Investigating the Effects of Prerequisite CS1 Options for a CS2 Course Through an Analysis of Student Project Scores in CS2

Dr. Laura K. Alford, University of Michigan

Laura K. Alford is a Lecturer IV at the University of Michigan. She researches ways to use data-informed analysis of students' performance and perceptions of classroom environment to support DEI-based curricula improvements.

Dr. James A. Juett, University of Michigan

James Juett is a member of the teaching faculty in CSE at the University of Michigan.

Heather Rypkema, University of Michigan

Heather Rypkema is Head of Learning Analytics at the Center for Research on Learning and Teaching (CRLT), as well as Assistant Director with the Foundational Course Initiative (FCI). Prior to shifting into the field of educational assessment and analytics, she was a faculty member at various institutions with research specializations in theoretical quantum chemistry and laser photochemistry.

Investigating the Effects of Prerequisite CS1 Options for a CS2 Course Through an Analysis of Student Project Scores in CS2

Abstract

This complete research paper details our analysis of the effect of students' pathways to a CS2 course on their project scores, exam scores, and final grades in the CS2 course.

Many institutions of higher learning have a standard multi-course sequence that covers the fundamentals of computer science/programming. The general introductory course is usually referred to as "CS1" in the literature; the second course usually covers data structures and is referred to as "CS2" in the literature.

A large CS2 course at the University of Michigan has four options for a student to meet the CS1 prerequisite: credit for a college-level CS1 course at the home institution, transfer credit for a college-level CS1 course at different institution, a diagnostic test for competency, and AP credit (score of 5) for placing out of the CS1 course prerequisite requirement. These prerequisite options can represent four fundamentally different paths for students to take, and we wished to empirically verify students pursuing each pathway are sufficiently prepared to succeed in CS2 and identify any salient difference between outcomes in CS2 dependent on pathway.

Our institution historically did not provide a mechanism for students to use AP credit to place out of CS1 as it was assumed that the high school AP courses did not sufficiently prepare students for the rigors of a CS2 course taken in their first semester in college. However, as more students enrolled with significant programming experience in high school, and as more students ceased to be challenged in the CS1 courses, the path for accepting AP credit was formalized starting in the Fall 2019 semester, such that students who earned a 5 on the AP test could enter CS2 directly. This research was motivated in part by a desire to analyze whether students who enter CS2 via the AP credit path struggle more than, less than, or about the same as students entering via a college-level CS1 course.

Our assessment of student success was based on programming project scores, exam performance, and overall letter grades across seven years' worth of data (13,000+ total students). Our results show that across all metrics, students entering via the AP pathway generally perform significantly better than students entering via our traditional CS1 courses. Students who receive transfer credit for a CS1 course perform comparably to students who took a CS1 course in-house. Both alternative pathways have particularly low failure rates.

Introduction and Motivation

The terms "CS1" and "CS2" are loosely used to refer to students' first and second courses in introductory computer science and programming, though there is wide variation among institutions and instructors in the precise meaning of these terms [1]. These courses are the topic of a significant body of literature, including research into teaching practices, assessment methodologies, student populations and experiences, and design choices such as modes of instruction, student engagement, or the programming language used [2, 3, 4].

Motivation to Investigate Pathways to CS2

The University of Michigan – Ann Arbor provides several pathways into CS2 (EECS 280), including three in-house CS1 courses (ENGR 101, ENGR 151, EECS 183), CS1 equivalency for students with AP or transfer credit, and a diagnostic process for direct placement into CS2. Allowing many, varied pathways provides flexibility for students and broadens opportunities, but this must be balanced against ensuring students who enter via each pathway are well-prepared to succeed in the CS2 course. Empirical analysis of student outcomes provides a critical supplement to instructors' intuition for determining whether this balance is met.

The primary in-house CS1 courses, ENGR 101, ENGR 151, and EECS 183, all cover fundamental programming concepts in C++ to prepare students to take EECS 280, but their approaches are fairly different. The first half of ENGR 101 covers basic MATLAB programming with the second half focusing on C++, with vectors of structs being the most complex concept taught. ENGR 151 covers the same concepts as ENGR 101 but at an accelerated pace allowing for deeper understanding and practice. EECS 183 teaches solely C++, with greater breadth and depth of coverage than ENGR 101, including an introduction to object-based programming with classes. The course instructors have always been cognizant of these differences and worry that the two approaches, while appropriate for the individual courses, may have unintended consequences for students in their later courses.

The AP credit pathway to CS2 is relatively new. In Fall 2019, the Computer Science and Engineering Division of the Electrical Engineering and Computer Science Department (EECS-CSE) approved a policy that students who scored 5 on the AP Computer Science A test could place into EECS 280, without requiring credit for a college-taught CS1 course or any additional evaluation. These students would have significant experience with Java and mastery of fundamental programming concepts (given an AP test score of 5), but would not have direct experience with C++, the language used in EECS 280. This, combined with a significantly higher workload than our CS1 courses, could make the transition into EECS 280 challenging for first-year students just entering college. Therefore, it is worth verifying the AP pathway sets most students up for success.

Students at the University of Michigan – Ann Arbor may also enter EECS 280 directly if they receive transfer credit for ENGR 101 or EECS 183 by taking a course from another institution that is approved for equivalency. However, a longstanding concern is that there is both high variability in external courses and limited ability for equivalency evaluators to determine whether students' level of preparedness is comparable to the in-house CS1 $\rightarrow$ CS2 pathway. While many transfer students will be well prepared, it is plausible that a significant group enter CS2 without

adequate preparation. In this case, action may be needed to 1) assist transfer students with determining readiness for EECS 280 and 2) provide additional resources and support for students in need.

Additional introductory programming courses at the University of Michigan – Ann Arbor have been recently developed for a wide array of student interests and needs, from robotics to programming for the sciences to programming for social justice. These courses may become new candidates for CS1 prerequisite approval, opening up even more potential pathways to EECS 280. Analyzing our existing pathways will help with future decision-making.

Related Works

A significant and increasing body of work concerns the prediction of student outcomes in introductory computer science courses [5, 6], including attempts to develop automated tools for predicting student performance (e.g. [7]). The categorization by Hellas et al. gives a coarse-grained picture of features that have been studies for the prediction student performance in CS courses: demographics, personality, prior academic performance, behavioral data, and institutional background [5].

In particular, performance in previous computer science courses or on examinations is a frequently studied and often effective predictor of success [8, 9, 10, 11, 12].

Relatively fewer studies have focused on the impact of different pathways to CS2. Ellis et al. found mixed results on the significance of the particular CS1 pathway taken by students (traditional CS1 vs. transfer credit) [8, 9]. Catanese et al. found transfer students had similar levels of success in the third-level computer science course as students who followed an in-house CS1 pathway [13]. Early access to CS education via AP Computer Science has been found to be correlated with improved performance in college CS courses across a variety of levels of the curricula [14].

Another potential factor is whether the programming language used in CS1 matches the one used in CS2. Rahman and Tyagi found that the programming language used in a prior course impacted retention from CS1 to CS2, but not performance in CS2 [15]. Enbody et al. report on a switch from C++ to Python in CS1 and conclude the Python course was just as effective in preparing students for CS2 in C++ [16, 17].

EECS 280 - CS2 at the University of Michigan – Ann Arbor

EECS 280 “Programming and Intro Data Structures” covers a fine-grained machine and memory model, complex programming concepts (e.g. polymorphism, dynamic resource management, generic programming, etc.), and a set of basic data structures and their implementation (e.g. unsorted vs. sorted arrays, linked lists, binary search trees). EECS 280 has also sometimes been called CS1.5, given students pursuing a CS major or minor will subsequently take EECS 281 “Data Structures and Algorithms” as well.

EECS 280 fulfills a core requirement for computer science, computer engineering, electrical engineering, data science, and robotics majors and an elective for several other programs. The course is also a core requirement for computing-related minors.

EECS 280 is taught in C++ and presumes familiarity with basic C++ programming (or that students will need to put in some extra initial work to get up to speed). There are also implicit dependencies on general pragmatics, such as the use of a C++ development environment and compiler toolchain - the course makes supplementary tutorials and office hours support available for students who may not be as comfortable with these. Nevertheless, Faculty, TAs, and students report anecdotal evidence that the initial ramp-up is challenging for some students.

Students in EECS 280 complete five major programming projects throughout the term, generally including a mix of scaffolded programming and open-ended design and implementation of a related application. Project submissions are automatically graded for implementation correctness, thoroughness of student-created test suites, and automated style checks. Students are evaluated on midterm and final exams. Roughly speaking, project submissions and exams account for the vast majority of students' overall grades. The median grade in the course is a B+. Students need a C or better to count the course toward program requirements or as a prerequisite for future courses. (Students receiving a C- still earn credit for the course, but cannot use it to fulfill requirements unless they retake it for a better grade.)

Methods

Our dataset includes over 13,000 students who took EECS 280 between Fall 2016 and Winter 2023 during a regular 4-month semester (Fall = Sep-Dec, Winter = Jan-Apr). We excluded summer offerings, which are at an accelerated pace. Complete data for Fall 2020 were not available, so we also excluded this term.

Assessment in EECS 280 has remained relatively consistent throughout the covered time period. The general distribution of letter grades is similar across terms. The programming projects in EECS 280 have not changed significantly since 2016, including the autograding mechanism used to evaluate student submissions (an exception is project 4, where the application *topic* has changed). In our analyses below, we compare student performance on projects and overall letter grades across all terms.

A change to curving policy affects numeric exam and overall scores in our dataset. Prior to Winter 2021, the course was curved by adjusting letter grade thresholds at the end of the course, so the exam and overall scores in are dataset are "pre-curve". Starting in Winter 2021, individual exams were curved instead, with letter grades assigned on a straight scale. This is done in a way that preserves the distribution of course letter grades, but means that exam and overall scores in our dataset are "post-curve" and appear higher for Winter 2021 and beyond. For this reason, our analyses below only consider exam and overall course scores for Winter 2021 and beyond.

Pathways to EECS 280

Several pathways to EECS 280 are available, including three in-house CS1 courses that satisfy the course prerequisite: ENGR 101, ENGR 151, and EECS 183. Students may also fill this requirement with transfer credit for an approved equivalent course from another institution, or with a score of 5 on the AP Computer Science A test. Finally, students may request permission to enroll without officially meeting prerequisite requirements based on a diagnostic project

submission and review of their background experience. More detail on each pathway is given here:

ENGR 101. Students in the College of Engineering are generally required to take ENGR 101 ("Introduction to Computers and Programming") during their first year, either in Fall or Winter term. Of these, approximately 42 percent go on to take EECS 280. The first half of the course covers an introduction to programming in MATLAB and the second half introduces C++.

ENGR 151. Students in the College of Engineering may elect to take ENGR 151 ("Accelerated Introduction to Computers and Programming"), which is taught at a faster pace and covers additional content beyond ENGR 101. ENGR 151 is usually only offered in the Fall term. Of students who take ENGR 151, approximately 77 percent go on to take EECS 280. Like ENGR 101, ENGR 151 covers both MATLAB and C++.

EECS 183. Students in many other schools and colleges (i.e. other than the College of Engineering) at the University of Michigan – Ann Arbor may take EECS 183 ("Elementary Programming Concepts"). While many students choose to take EECS 183 relatively early in their undergraduate career, this is not required. EECS 183 is offered in both the Fall and Winter terms. Of students who take EECS 183, approximately 48 percent go on to take EECS 280. EECS 183 covers programming in C++ with a small amount of Python at the end of the semester.

TRANSFER. Students may take EECS 280 directly if they have transfer credit for a prerequisite course, generally ENGR 101 or EECS 183. Equivalency for ENGR 101 is rare, since few external courses cover both MATLAB and C++ (students transferring CS1 in a different language earn ENGR 101x credit, which does not fulfill the EECS 280 prerequisite). Equivalency for EECS 183 is granted to students who have taken an intro programming course in C or C++. In all cases, a credit evaluator attempts to determine whether material coverage and level of rigor are comparable to CS1 courses at the University of Michigan – Ann Arbor.

AP. Students who take AP Computer Science A [18] in high school and earn a 5 on the AP exam are eligible to earn transfer credit for EECS 180 ("Exam/Transfer Introductory Computer Programming Credit") and place directly into EECS 280. No additional evaluation is required. According to The College Board, AP CS A is intended as an "equivalent to a first-semester, college-level course in computer science". AP CS A covers programming in Java.

DIAGNOSTIC TEST. Students not meeting the requirements other pathways may request exceptional permission to enroll directly in EECS 280 without meeting the course prerequisites. This process involves student submission of an autograded diagnostic C++ project with a score of 90% or better, in addition to review of the student's previous CS coursework and other programming experience. In general, the diagnostic process is designed to evaluate whether students are well-prepared to begin a second-level C++ programming course.

Analysis

This analysis used the gradebook data from EECS 280 from Fall 2016 – Winter 2023. In order to authentically ascertain the impact of pathway, the data were filtered to specifically include the earliest term students took EECS 280, regardless of pathway or other factors. University of Michigan – Ann Arbor courses students took to qualify for EECS 280 were flagged as their

pathway, while other transfer credits (via courses at other institutions, diagnostic tests, or AP credit) were assigned accordingly.

Statistical analyses and visualizations were conducted in R. Out of more than 13,000 student records, about 300 had no pathway data in the student database and thus were removed from the analysis. Roughly 20 students in the course gradebook could not be connected to the student database and were also eliminated from this analysis. The remaining students (more than 12,800) and their overall, exam, and project grades were included in this analysis. It was validated that enrollment pathway data did not pre-date EECS 280 enrollment, ensuring that pathway designations did not predate the dataset.

Results

The percentage of students in EECS 280 per pathway is shown in Fig. 1. There are more than 12,800 students represented in this chart, so even the 1% DIAGNOSTIC TEST pathway category represents over 100 students.

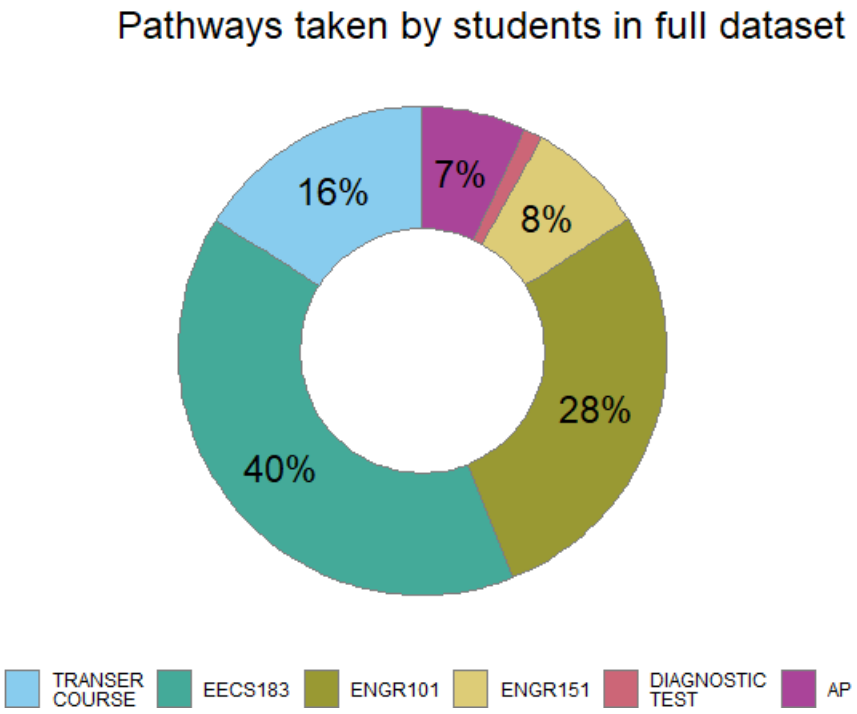

Figure 1: Unsurprisingly, the majority of students in the CS2 course come from the two largest in-house CS1 courses. However, the other pathways all have sufficient percentage to warrant consideration in this analysis.

Mean Project Score for Each Student. The mean score across all five projects, for all students in the dataset, is shown in Fig. 2. The distribution of the mean project score of the students in each pathway is shown along with data points for each individual student in the pathway.

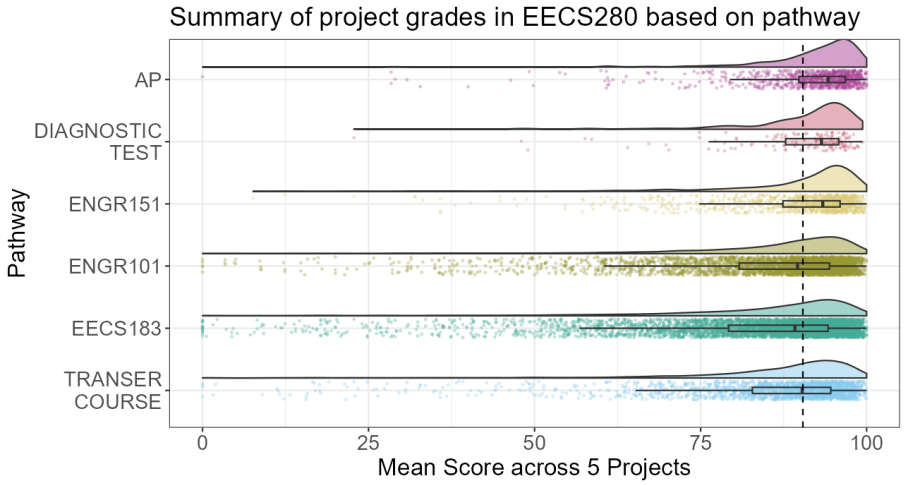

Figure 2: This chart shows the mean project score for each student in CS2 (each student is a dot in the chart), along with a distribution to visualize the differences between the pathways. The majority of students do relatively well on the projects overall, but each pathway has a long tail, indicating that individual students in any pathway may struggle in any given semester. The vertical dashed line represents the overall median score.

Mean Score for Each Project. The mean scores on each project, for all students in the dataset, are shown in Fig. 3. This figure shows the difference in the mean scores of the students in each pathway for each of the five projects.

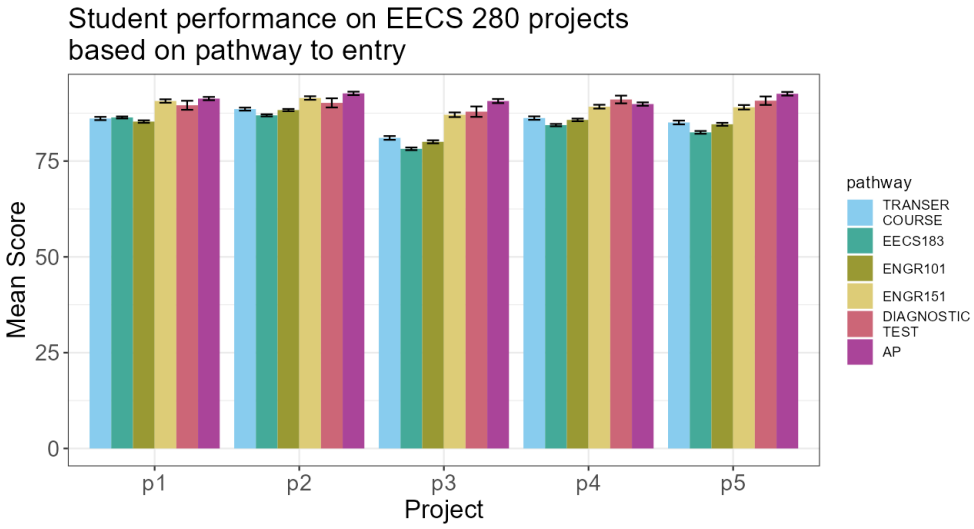

Figure 3: This chart shows the mean score for each project. The overall mean project score is useful, but this shows the difference between the projects. As noted later, the projects have different levels of complexity and scope.

Mean Score for Each Project with Standard Error. The mean scores on each project, for all students in the dataset, with standard error are shown in Fig. 4. This figure more clearly shows the difference in the mean scores of the students in each pathway for each of the five projects.

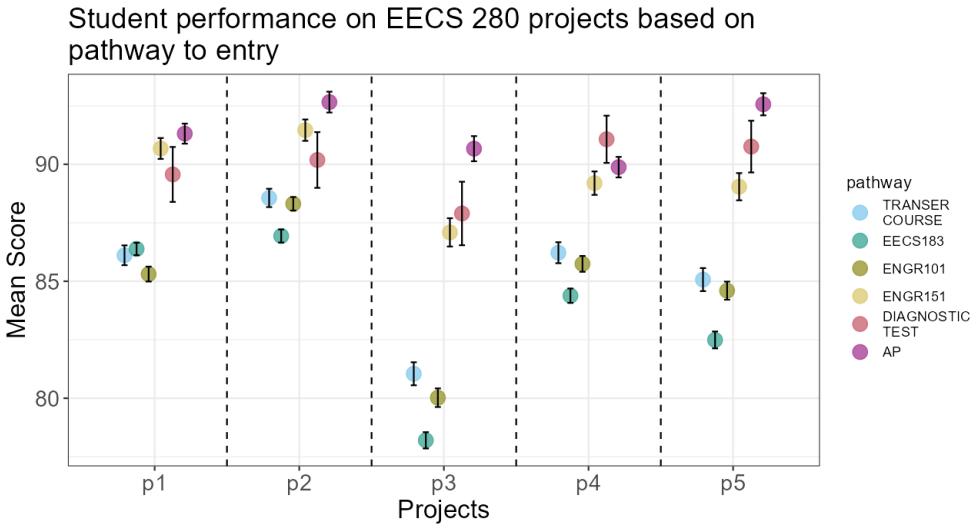

Figure 4: This chart shows the mean score on each project for students in the different pathways. The standard error is shown for each data point. Note that the vertical scale does not start at zero so as to better show the differences between the mean scores of the different pathways.

Final Letter Grades. The final letter grades, for all students in the dataset, are shown in Fig. 5. The overall letter grade distribution in the course as a whole has remained consistent for many years.

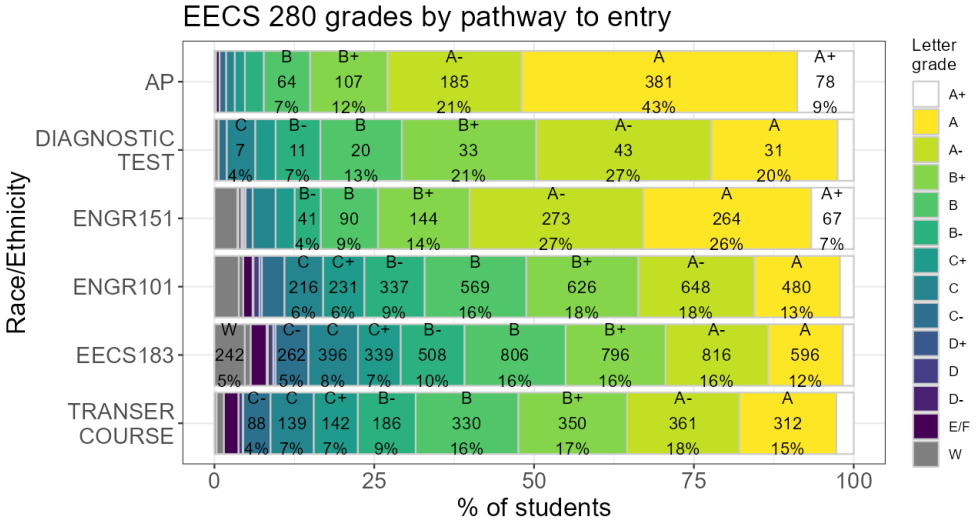

Figure 5: This chart shows the distribution of the final course letter grades for students in the different pathways.

Mean Exam Scores and Overall Mean Score in Course. The mean exam scores and mean overall score in the course, for students in the Winter 2021 – Fall 2023 terms, are shown in Fig. 6. Recall that we are limiting this analysis to this subset of data due to the change in exam grading policy, which in turn affects the overall course scores (as opposed to the letter grades, which were previously curved).

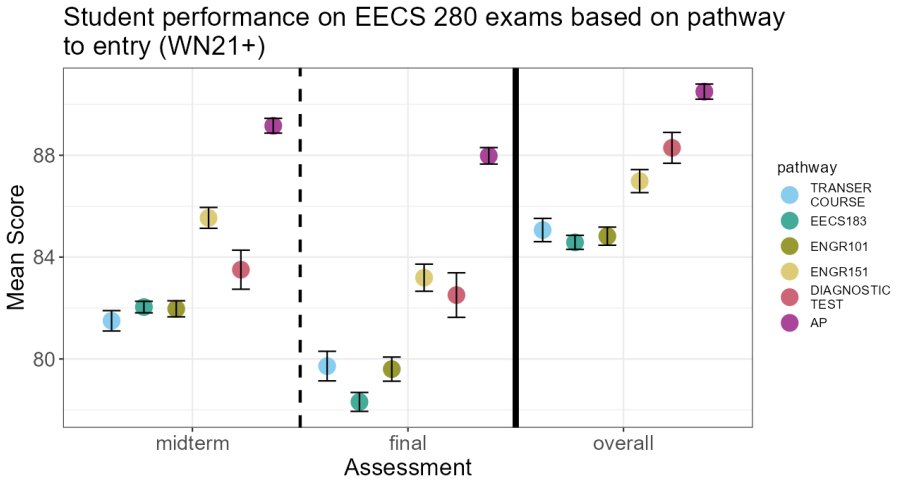

Figure 6: This chart shows the mean exam scores and mean final course scores for students that took the course in the Winter 2021 – Fall 2023 terms (a subset of the entire dataset). The standard error is shown for each data point.

Overall Score in Course. The overall scores in the course, for students in the Winter 2021 – Fall 2023 terms, are shown in Fig. 7. The distribution of the overall course score of the students in each pathway is shown along with data points for each individual student in the pathway.

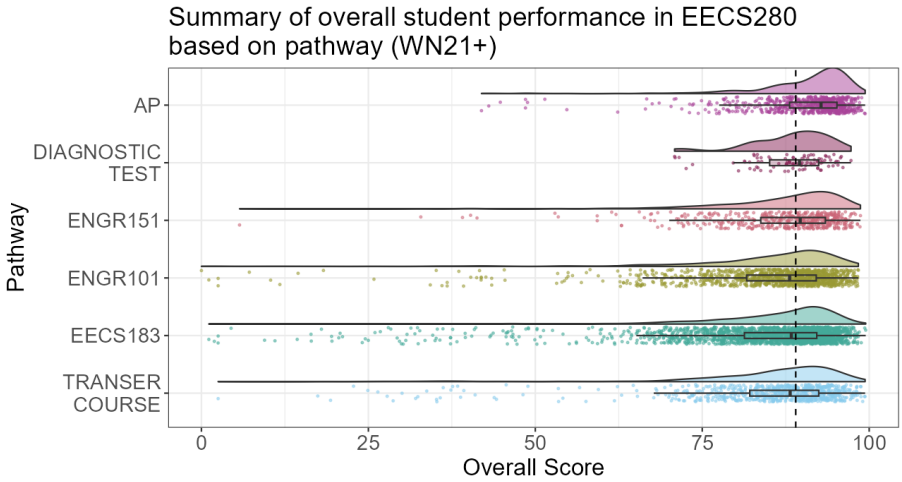

Figure 7: This chart shows the final course scores for students that took the course in the Winter 2021 – Fall 2023 terms (a subset of the entire dataset; each student is a dot in the chart), along with a distribution to visualize the differences between the pathways. The vertical line indicates the overall median.

Discussion

In general, students entering via the ENGR 151, AP, and DIAGNOSTIC TEST pathways show the strongest performance in EECS 280. This is expected for the ENGR 151 pathway, since those are students who self-select into the accelerated intro CS course. Additionally, students who enter via DIAGNOSTIC TEST are generally those that did not happen to have transfer credit but are nevertheless exceptionally well-qualified (as vetted by the diagnostic evaluation).

The data alleviate the concern that students entering via the AP pathway are under-prepared, either because they don't have previous experience with C++ or because they take CS2 as a first CS college course. In fact, the AP pathway shows the strongest performance of any group in our analysis. The D/F/W rate is negligible. We emphasize this result applies only for students who took the AP CS A course *and* also earned a 5 on the AP test, which are the criteria for this pathway. Is not necessarily surprising these students are highly successful, given that CS1 performance is well-established in the literature as an effective predictor of CS2 performance.

Indeed, our data support strong CS1 experience as an effective predictor of success in CS2 even when the programming languages used are different (i.e. Java for AP CS A vs. C++ for EECS 280). This suggests possible additional pathways to CS2 could be considered, perhaps through demonstrated excellent performance in a broader range of CS1 courses than our current TRANSFER pathway allows, or via a more generic assessment than our C++ DIAGNOSTIC TEST, such as the language-independent FCS1 [19, 20].

Students entering via the TRANSFER pathway have similar performance to those taking in-house CS1 courses, both in aggregate score metrics and in the distribution of final letter grades, with notably lower D/F/W rates. Thus, the data suggest students who enter via TRANSFER are not at elevated risk compared to other pathways.

Project 3 in EECS 280 is considered by instructors and students to be the most challenging project with the highest workload, followed by Project 5. For these projects, the data suggest greater variation between pathway on more challenging projects.

Project 1 is designed with a significantly smaller workload than the others and intended to act as a "ramp-up", yet we found variation in performance comparable to projects 2 and 4. This may suggest the differences between pathways are more pronounced right at the start of the term.

It is interesting to note that D/F/W rates are lower for all the alternative pathways (AP, DIAGNOSTIC TEST, TRANSFER) than for the traditional CS1 courses, including the accelerated ENGR 151. This may suggest particular resilience among students who pursue an alternative pathway.

Limitations

The populations of students entering via the AP, DIAGNOSTIC TEST, and TRANSFER pathways are all subject to significant filter effects. In the case of AP credit and the DIAGNOSTIC TEST process, students must score highly enough on an evaluation to qualify for entry. TRANSFER students generally enter the University at large via a different process and

generally take our CS2 with more previous college experience. While we can assess that students entering via these pathways are generally well-prepared, further work would be necessary to determine how much of this is due to the selectivity of those pathways or the correlation of those pathways to other privilege (e.g. access to AP CS in high school). Additional work is also needed to assess potential impacts of widening those pathways (e.g. only requiring a 4 on the AP CS A test).

Our dataset does not include students who initially enrolled in EECS 280 but dropped within the first 3 weeks of the course, either because they found it difficult to keep up or for some other reason. This may be particularly relevant for the TRANSFER pathway, where it is possible a some students are not well-served by their placement directly into EECS 280 and silently switch to one of the CS1 intro courses.

Our present work does not attempt to evaluate the quality of individual courses or pathways. Our main in-house CS1 courses represent different populations of students and are subject to filter effects. Only College of Engineering students enter via the ENGR 101/ENGR 151 pathways, whereas students from many different colleges/schools enter via EECS 183. These each have their own admissions processes and criteria. ENGR 101 and ENGR 151 represent a partition of students based on who self-selects into the "accelerated" version. Students in EECS 183 focus primarily on C++, whereas students in the ENGR 101 and ENGR 151 courses also learn MATLAB.

We found that students who performed well on the AP CS A test were well-prepared to succeed in EECS 280 despite no requirement of previous C++ experience. However, these students have a foundation in Java, which is similar to C++ in several ways. Both belong to the imperative paradigm – it is possible students coming from e.g. CS1 taught in a functional language may have a different experience. Or, languages such as Python that are less syntactically similar to C++ may present different challenges.

Conclusions and Recommendations

We conclude that our present alternative pathways to enter EECS 280 generally serve students well. The data show students who score a 5 on the AP CS A test are well qualified to enter EECS 280 directly and perform exceptionally well in the course. Students with TRANSFER credit for one of our CS1 courses generally perform comparably with students who took one of the actual in-house CS1 offerings. Students entering via alternative pathways have lower D/F/W rates than traditional CS1, even when compared to the accelerated ENGR 151.

Strong performance in the AP pathway also suggests a same-language CS1 offering is not critical for students who have performed well in their previous CS coursework. Therefore, we recommend that institutions consider approving CS1 courses as prerequisites for CS2 based on the general computing concepts covered in the CS1 course without penalized the CS1 course based on the programming language taught.

Future Work

The findings presented in this paper suggest several avenues for future research that can benefit the education community.

First, the scope of this work did not include a detailed breakdown of the demographics within each pathway. We recommend that a follow-up study be conducted that especially looks at the intersectionality of various demographic groups within each pathway to determine if there are underlying biases that should be mitigated.

Second, the question of "What language should we teach our course in?" is asked of CS1 and CS2 instructors all the time. Courses can and do change their language(s) of choice based on the changing needs of students. While the findings here suggest that a change in language may not create many challenges to students in downstream classes, it is important to recall the limitations of this study. Many students have self-selected into pathways that indicate they are strongly prepared to pick up a new language without much trouble. Not all students will be able to do this with ease. Nor are computing languages structured in the same way. An AP student with a background in Java may transition better to a course in C++ than a CS1 student with a background in Python... or they might be fine. Additional studies including a wider variety of CS1 pathways to CS2 would be needed to determine the impact of such curricular changes.

Finally, we recommend that institutions that are considering broadening their approved CS1 $\rightarrow$ CS2 pathways regularly assess student performance in CS2 through a variety of assessment methods (projects, exams, final grades, etc.) to track student success.

Furthermore, we acknowledge that success as measured by assignment scores or letter grades do not tell the full story of student experiences in EECS 280. We did not consider the amount of time required to complete assignments, nor the weight of students' other courseload while taking EECS 280, which may be generally different for different pathways. We also did not investigate students' future pursuits after taking the course. Students' qualitative experiences, including attitude toward the course, sense of belonging, and interaction with others are all important components of success as well. Future studies investigating these avenues of research will provide valuable insight into student success.

Acknowledgement

This research was supported by the Foundational Course Initiative at the University of Michigan – Ann Arbor under IRB HUM00150716.

References

[1] M. Hertz, "What do" cs1" and" cs2" mean? investigating differences in the early courses," in *Proceedings of the 41st ACM technical symposium on Computer science education*, 2010, pp. 199–203.

[2] B. A. Becker and K. Quille, "50 years of cs1 at sigcse: A review of the evolution of introductory programming education research," in *Proceedings of the 50th acm technical symposium on computer science education*, 2019, pp. 338–344.

[3] R. P. Medeiros, G. L. Ramalho, and T. P. Falcão, "A systematic literature review on teaching and learning

introductory programming in higher education," *IEEE Transactions on Education*, vol. 62, no. 2, pp. 77–90, 2018.

[4] A. Luxton-Reilly, Simon, I. Albluwi, B. A. Becker, M. Giannakos, A. N. Kumar, L. Ott, J. Paterson, M. J. Scott, J. Sheard *et al.*, "Introductory programming: a systematic literature review," in *Proceedings companion of the 23rd annual ACM conference on innovation and technology in computer science education*, 2018, pp. 55–106.

[5] A. Hellas, P. Ihantola, A. Petersen, V. V. Ajanovski, M. Gutica, T. Hynninen, A. Knutas, J. Leinonen, C. Messom, and S. N. Liao, "Predicting academic performance: a systematic literature review," in *Proceedings companion of the 23rd annual ACM conference on innovation and technology in computer science education*, 2018, pp. 175–199.

[6] K. Quille and S. Bergin, "Cs1: how will they do? how can we help? a decade of research and practice," *Computer Science Education*, vol. 29, no. 2-3, pp. 254–282, 2019.

[7] ——, "Programming: predicting student success early in cs1. a re-validation and replication study," in *Proceedings of the 23rd annual ACM conference on innovation and technology in computer science education*, 2018, pp. 15–20.

[8] M. Ellis and S. Hooshangi, "Replication and expansion study on factors influencing student performance in cs2," in *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, 2023, pp. 896–902.

[9] S. Hooshangi, M. Ellis, and S. H. Edwards, "Factors influencing student performance and persistence in cs2," in *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education-Volume 1*, 2022, pp. 286–292.

[10] H. Danielsiek and J. Vahrenhold, "Stay on these roads: Potential factors indicating students' performance in a cs2 course," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 2016, pp. 12–17.

[11] L. Layman, Y. Song, and C. Guinn, "Toward predicting success and failure in cs2: A mixed-method analysis," in *Proceedings of the 2020 ACM Southeast Conference*, 2020, pp. 218–225.

[12] L. Beck, P. Kraft, and A. W. Chizhik, "Predicting student success in cs2: A study of cs1 exam questions," in *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education-Volume 1*, 2022, pp. 140–146.

[13] H. Catanese, C. Hauser, and A. H. Gebremedhin, "Evaluation of native and transfer students' success in a computer science course," *ACM Inroads*, vol. 9, no. 2, pp. 53–57, 2018.

[14] C. Alvarado, G. Umbelino, and M. Minnes, "The persistent effect of pre-college computing experience on college cs course grades," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018, pp. 876–881.

[15] F. Rahman and J. Tyagi, "Investigating the role of different prep pathways on cs2 performance across three different majors," in *Proceedings of the 22nd Annual Conference on Information Technology Education*, 2021, pp. 141–146.

[16] R. J. Enbody, W. F. Punch, and M. McCullen, "Python cs1 as preparation for c++ cs2," in *Proceedings of the 40th ACM technical symposium on Computer Science Education*, 2009, pp. 116–120.

[17] R. J. Enbody and W. F. Punch, "Performance of python cs1 students in mid-level non-python cs courses," in *Proceedings of the 41st ACM technical symposium on Computer science education*, 2010, pp. 520–523.

[18] College Board, *AP Computer Science A: Course and Exam Description*, 2023, accessed: 2/7/2024. [Online]. Available: https://apcentral.collegeboard.org/media/pdf/ap-computer-science-a-course-and-exam-description.pdf

[19] A. E. Tew and M. Guzdial, "The fcs1: a language independent assessment of cs1 knowledge," in *Proceedings of the 42nd ACM technical symposium on Computer science education*, 2011, pp. 111–116.

[20] M. C. Parker, M. Guzdial, and S. Engleman, "Replication, validation, and use of a language independent cs1 knowledge assessment," in *Proceedings of the 2016 ACM conference on international computing education research*, 2016, pp. 93–101.