Computer Science Department
Programmatic Self-Study

April 2023

Preface **3**

1. Program Context and Curriculum **4**

1.1 Introduction 4

1.2 Program Mission and Goals 4

1.3 Program Learning Outcomes 5

1.4 Computer Science at SSU 6

1.4.1 Program Strengths and Distinctiveness 8

1.5 Curriculum & Assessment 9

1.5.1 Sample four-year and two-year plans 10

1.5.2 Alignment of Curriculum with PLOs 11

1.6 Interaction with Other Programs 14

1.7 Reflections on Prior Program Review 14

1.8 Changes from Prior Program Review 15

1.9 National Trends in Computing Workforce and Education 16

1.9.1 Workforce Trends 16

1.9.2 Trends in CS Major Enrollment 16

1.9.3 The “Capacity Crisis” 18

1.9.4 Faculty Recruitment and Diversity Challenges 23

2. Assessment **26**

2.1 Direct Assessment and Assessment of PLOs 26

2.2 Indirect Assessment Findings 29

2.2.1 Exit Survey, AY 2015 30

2.2.1 Exit Interview, Spring 2022 31

2.2.2 Alumni Survey, Fall 2022 32

2.3 Assessment Reflection 32

2.4 Assessment-Driven Curricular Changes 32

2.5 Future Assessment Plans 33

3. Faculty **35**

3.1 Composition 35

3.2 Faculty Workload 40

3.3 Faculty Specializations 46

3.4 Professional Development 49

3.5 Scholarship and Creative Activities 51

4. Program Resources **52**

4.1 Student Support 52

4.1.1 Advising 52

4.1.2 Campus Support Services 53
4.1.3 Student research and/or community engagement 55
4.2 Library and Information Resources 56
4.3 IT Resources 57
4.3.1 Technology resources for classroom instruction 57
4.3.2 Technology resources for student work 57
4.3.3 Technology resources for course development and delivery 58
4.3.4 Technology resources for advancement of instruction / research / scholarship 58
4.4 Facilities and Instructional Spaces 59
4.4.1 Spaces supporting CS Instruction 59
4.4.2 Faculty offices & meeting spaces 62
4.4.3 Research & scholarship facilities 62
4.4.4 Student study & social spaces 62
4.5 Staff Support 63
5. Student Success 65
5.1 Student Enrollment and Demographics 65
5.1.1 Student Demographics: Gender 67
5.1.2 Student Demographics: Ethnicity 69
5.1.3 First Generation CS Majors 71
5.1.4 Student Demographics: Age 72
5.1.5 Unit Loads of CS Majors 73
5.2 Student Demand 75
5.3 Retention and Graduation 77
5.3.1 Retention Trends 80
5.3.2 Time to Degree 82
5.3.3 Degrees conferred 89
5.3.4 DWF Rates 90
5.3.5 Challenges of teaching combined courses 94
5.4 Alumni Feedback and Outcomes 95
6. Reflection and Plan of Action 98
6.1 Retain a Diverse Faculty Body 98
6.2 Continue to Support our Diverse Student Body 99
6.3 Plan an Overdue Curricular Redesign 100
6.4 Align Program Growth with Institutional Resources 101
6.5 Address Challenges in Staffing and Scheduling 101
Bibliography 103
Appendices 107

Preface

This self-study is a synopsis of the Computer Science program at Sonoma State University. In addition to documenting our program, students, faculty and resources, it addresses the educational effectiveness of our program and thoughts on its future. The self-study is supplemented by many supporting documents, grouped as appendices.

1. Program Context and Curriculum

1.1 Introduction

Computer Science (CS) is the scientific study of computing devices, the software that drives them, and the computational tasks they are capable of performing. As such, computer science includes both hardware science and software science; and as with all sciences, each of these have both theoretical and applied components. Computing theory shares knowledge and techniques with the fields of mathematics, physics, engineering, philosophy, psychology, and linguistics. Its applications span the range of human endeavors: the physical, life, and social sciences; the literary, visual, and performing arts; law, government, recreation, and virtually every sector of the commercial world. Thus computer science is by its very nature an interdisciplinary subject that offers both a solid, unifying foundation for a liberal arts education and valuable career skills.

As a discipline, CS has one foot in the liberal arts tradition—with deep roots in mathematics and interdisciplinary connections to the physical sciences, social sciences, and arts—and another in the pre-professional domain, with a strong labor market for our majors and deep ties to electrical engineering. At Sonoma State, our CS faculty value and celebrate this dual nature: the pre-professional slant attracts a diverse group of career-oriented traditional and non-traditional students, and we routinely see students' family trajectories changed by the career opportunities open to them upon graduation, contributing to the CSU mission of economic mobility. As teacher-scholars, we also value the fact that this professional preparation consists of intellectually rigorous work within the liberal arts tradition and with connections to our colleagues' work in many disciplines across campus.

1.2 Program Mission and Goals

The mission of the Department of Computer Science is to offer expertise in the academic discipline of CS to our students, to the campus, and to the community. Foremost, we equip our undergraduate majors with a foundation in CS that is aligned to international standards and prepares them for success in technology careers. Our curriculum is delivered by experts in the field employing inclusive teaching methods and proven technologies, in a small-enough classroom setting that fosters community among students and faculty. Analytical problem solving is at the core of a computer science education. Students are expected to learn the mathematical foundations of computer science, the practice of software development in multiple programming languages individually and collaboratively, the intricacy of the architecture of large-scale software systems, the level at which the software and hardware systems interact down to the hardware architecture, and the application areas of computer science such as computer graphics, machine learning, artificial intelligence, and gaming.

The department implements practices that promote access, equity, and justice in our community, serving our professional missions as a Hispanic Serving Institution, a member of the

Computing Alliance of Hispanic-Serving Institutions, and a member of the National Center for Women & Information Technology. We enable student and faculty scholarship and engagement with professional communities at the regional, state, national, and international levels, and we provide meaningful connections among students, alumni, local industry, and other academic institutions.

1.3 Program Learning Outcomes

The program learning outcomes (PLOs) for a Bachelor of Science in Computer Science at Sonoma State University are:

1. Apply software design and engineering principles to develop and evaluate a computing-based solution to meet a given need.
2. Apply theoretical foundations, algorithmic principles, and computer organization fundamentals to analyze the tradeoffs involved in designing computer-based solutions.
3. Select appropriate tools and techniques for a given computing task, and quickly develop proficiency with new tools.
4. Collaborate and communicate effectively with others to accomplish professional goals.
5. Drawing on the foundations of a strong liberal arts education, make informed ethical judgments grounded in social and professional responsibility.

These outcomes highlight the academic emphasis of our program and also acknowledge its context as a part of the CSU—featuring a strong lower- and upper-division GE curriculum, with a mandate for modular and transferable curricula—and as California's member of the Council of Public Liberal Arts Colleges (COPLAC).

Our program learning outcomes also complement the GE program to fulfill the Western Association of Schools and Colleges (WASC) core competencies for undergraduate programs [WASC21]. Table 1.1 shows their alignment with the four mandatory WASC core competencies.

Table 1.1 Alignment of SSU CS PLOs with WASC core competencies

SSU CS PLO	**Relevant WASC CCs**
SSU-1 Apply software design and engineering principles to develop and evaluate a computing-based solution to meet a given need.	● Quantitative Reasoning ● Critical Thinking ● Written Communication
SSU-2 Apply theoretical foundations, algorithmic principles, and computer organization fundamentals to analyze the tradeoffs involved in designing computer-based solutions.	● Quantitative Reasoning ● Critical Thinking
SSU-3 Select appropriate tools and techniques for a given computing task, and quickly develop proficiency with new tools.	● Information Literacy ● Critical Thinking

SSU-4 Collaborate and communicate effectively with others to accomplish professional goals.	● Written and Oral Communication
SSU-5 Drawing on the foundations of a strong liberal arts education, make informed ethical judgments grounded in social and professional responsibility.	● Written and Oral Communication ● Critical Thinking

Further, we believe these program learning outcomes to be congruent with those outcomes embraced by peer CSU programs. In particular, the most common disciplinary accreditor for computer science undergraduate programs is ABET, under the ABET Computing Accreditation Commission (ABET-CAC). See Appendix E for alignment of our program learning outcomes aligned with the 2023-24 ABET-CAC criteria for accrediting computing programs.

1.4 Computer Science at SSU

Sonoma State University is organized into six schools. Computer Science is one of nine departments in the School of Science and Technology (SST). The computer science program was started in 1982, as a specialization within the mathematics department where it resided until 1986, after which it became a separate, independent department.

The Computer Science Department offers a standard curriculum of required and elective courses leading to a B.S. degree in Computer Science. The department also offers a CS minor. The department coordinates with other programs like mathematics and electrical engineering to offer pathways to help students to more reasonably be, for example, an Electrical Engineering major with a CS minor, a CS major with an EE minor, a CS major with a math minor, pursue a disciplinary mathematics concentration with an emphasis in CS courseware.

The Computer Science department has undergone many changes over the past four decades. Between the mid-1980s and mid-1990s, the department saw steady and continuous growth. It grew to serve ~300 major FTEs. This growth upturned steeply during the dot-com era (mid-1990s to early-2000s), when being a computer science major was perceived as a quick path toward a financially advantageous career. When the dot-com bubble burst, the number of majors shrank. Enrollment in the first course in the major (CS 115) went from a high of ~100 to a low of ~30. During the six academic years from 2000-2001 to 2005-2006, the computer science department averaged 144 major FTES. These enrollment trends were not unique to the CS department at SSU, but rather reflective of endemic, nationwide-phenomenon related to business and industry, and uncorrelated to the quality of a program or the reputation of the institution offering such programs.

Over the past few years, our department and program have been buffeted by both the pandemic and a decrease in the college-age population, like much of the higher education sector. Indeed, the principal challenge facing SSU as a whole is the existential budgetary shortfall caused by recent enrollment declines [Murphy22]. Over the period of this review, our program has faced an additional existential challenge that may be unique to computer science: a dramatic surge in

student demand that has overwhelmed our resources and threatened what modest gains we have made in reaching historically underrepresented populations. This growth is reflected in the number of CS students involved in the program (Figure 1.2), and its recent decline follows SSU's total enrollment decline (Figure 1.3). The challenges faced by our department should, of course, be viewed in the context of national trends in CS education—workforce trends, a "capacity" crisis, equity challenges, faculty retention challenges (see Section 1.8)—not exclusively in the context of college enrollment decline during the pandemic.

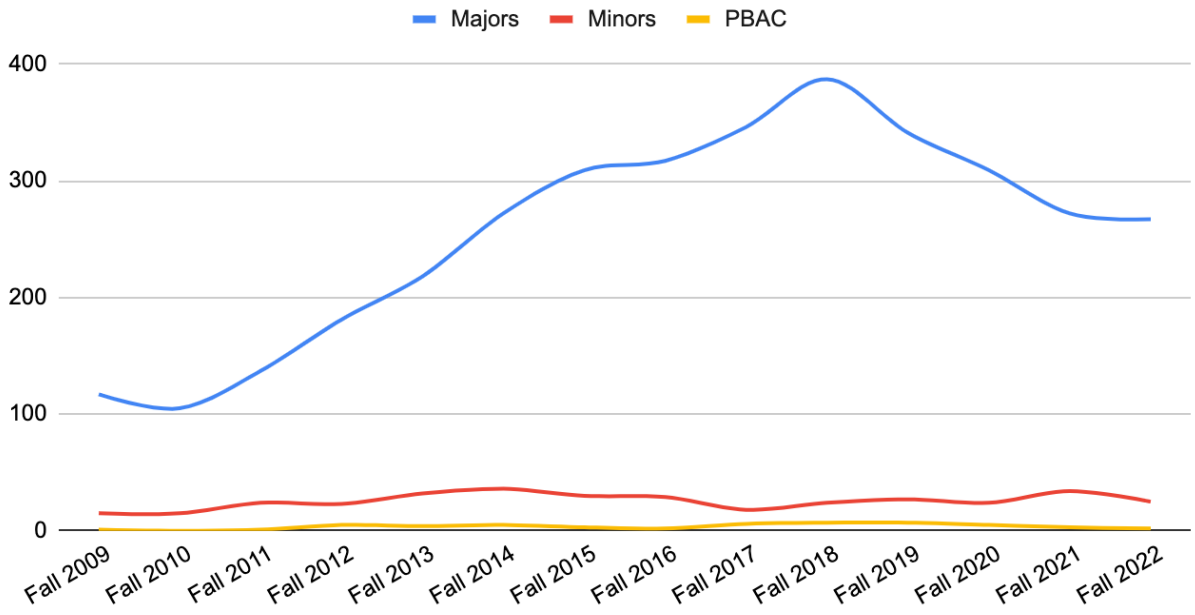

Figure 1.2. Total SSU CS Student Population, 2009–2022

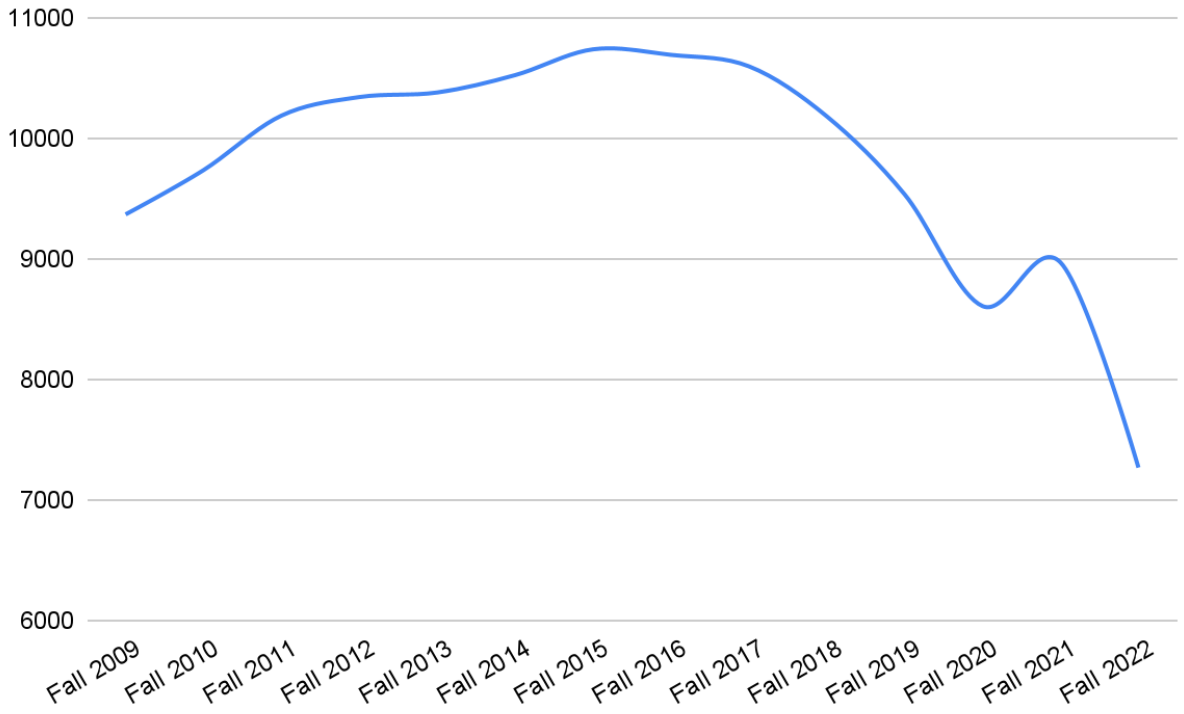

Figure 1.3. Total SSU Undergraduate Student Population, 2009–2022

1.4.1 Program Strengths and Distinctiveness

As the only public primarily undergraduate institution in the region, we serve an important state and regional need with local junior colleges and their CS transfer population (see Section 5.3). Instructor-supervised laboratory experience is essential to the quality of our program and distinguishes it from others. This experience prepares our students for their professional careers in an experience aligned with our identity as a high-touch liberal arts college. In spite of our student-to-faculty ratio, we have continued to engage in student-centered research and other high-touch activities. The program has a particular strength in platform-based development (e.g., iOS development) and full-stack development that has made many of our graduates competitive for employment in north bay industries; we have a moderately-sized cohort of students employed with mFoundry / FIS mobile and Disney+. Historically, the CS program at SSU has also been a national trendsetter in computer security, being one of the first universities to offer a “malware development” course. Its proximity to industry in the Bay Area and North Bay helps fulfill the near-endless demand for technology professionals in every commercial sector (transportation, education, communication, retail, entertainment, healthcare, financial, etc).

1.5 Curriculum & Assessment

The Bachelor of Science in CS degree requires 120 units for graduation. The major core requirements, upper-division elective options, capstone requirement, and support course options are given in Table 1.4 (see Appendix D for full course descriptions). Our program adheres to the SSU CS lower division transfer pattern, supporting the intent to allow community and junior college students to get a B.S. in computer science in two years after completing all lower-division requirements.

Table 1.4. Requirements for the B.S. in CS

Major Core Requirements (49 units)	• CS 115 - Programming I • CS 210 - Introduction to Unix • CS 215 - Programming II • CS 242 - Discrete Structures for Computer Science • CS 252 - Introduction to Computer Organization • CS 315 - Data Structures • CS 351 - Computer Architecture • CS 355 - Database Management Systems Design • CS 370 - Software Design and Development • CS 415 - Algorithm Analysis • CS 450 - Operating Systems • CS 454 - Theory of Computation • CS 460 - Programming Languages
Computer Science Electives (9 units) No more than 3 units can be satisfied by a combination of CS 349, CS 390, CS 495, and CS 497	9 units of the following: • CS 330 - Introduction to Game Programming • CS 340 - Computer Security and Malware • CS 349 - Problem Solving in a Team Environment • CS 360 - Object-Oriented Programming • CS 365 - Computer Networking and the Internet • CS 375 - Computer Graphics • CS 385 - Selected Topics in Computer Science • CS 386 - Selected Topics in CS with Lab • CS 390 - Computer Science Colloquium • CS 425 - Parallel Computing • CS 452 - Compiler Design and Construction • CS 465 - Data Communications • CS 480 - Artificial Intelligence • CS 495 - Special Studies • CS 497 - Internship
CS Capstone Requirement (3 units)	One of the following: • CS 470 - Advanced Software Design Project • CS 496 - Senior Research Project

Mathematics Support courses (10 units)	• MATH 161 - Differential and Integral Calculus I Two additional courses from the following: • MATH 165 - Elementary Applied Statistics • MATH 211 - Differential and Integral Calculus II • MATH 222 - Elementary Applied Linear Algebra • MATH 241 - Linear Algebra with Applications in Differential Equations • MATH 306 - Number Theory • MATH 316 - Graph Theory and Combinatorics • MATH 352 - Numerical Analysis • MATH 416 - Graph Theory and Combinatorics • MATH 430 - Linear Systems Theory • MATH 470 - Mathematical and Statistical Modeling • PHYS 214 - Introduction to Physics II

1.5.1 Sample four-year and two-year plans

The CS program prerequisite structure is fairly “vertical” (see the B.S. CS program prerequisite structure in Appendix E). As such, properly advising students in the major early in their college career is imperative to maintaining progress to graduation in a timely manner. A typical 4-year plan for first-time freshman (FTF) who are eligible to enroll in college-level math and English courses (GE MATH and ENGL eligible) is given in Table 1.5. A typical 5-year plan for FTF students who might not be GE MATH or ENGL eligible, or students who choose CS as a major in their second year at SSU is given in Table 1.6. The 2-year plan for transfer students who have completed the equivalents of all 100- and 200-level Computer Science and Support courses is given in Table 1.7.

Table 1.5. A possible 4-year plan for FTF who are GE MATH & ENGL eligible.

Year	Fall	Spring
1	CS 115 (4 units) MATH 161 (4)	CS 210 (1) CS 215 (4) CS 242 (4)
2	CS 315 (4) CS 370 (4) MATH Support (3-4)	CS 355 (4) CS Elective (3) CS 252 (4)
3	CS 415 (4) CS 351 (4) MATH Support (3-4)	CS 454 (4) CS 450 (4)
4	CS 460 (4)	CS Elective (3)

	CS Elective (3)	CS 470 (3)

Table 1.6. A possible 5-year plan for FTF who are not GE MATH & ENGL eligible.

Year	Fall	Spring
1	GE MATH and/or ENGL courses	GE MATH and/or ENGL Courses
2	CS 115 (4) MATH Support (3-4)	CS 210 CS 215 CS 242
3	CS 315 (4) CS 252 (4) CS 370 (4)	CS 351 (4) CS 355 (4) MATH Support (3-4)
4	CS 454 (4) CS 450 (4)	CS Elective (3) CS 415 (4)
5	CS Elective (3) CS 460	CS 470 (3) CS Elective (3)

Table 1.7. A possible 2-year plan for transfer students.

Year	Fall	Spring
1	CS 315 (4) CS 370 (4) CS 355 (4)	CS 415 (4) CS 351 (4) CS Elective (3)
2	CS 454 (4) CS 450 (4) CS Elective (3)	CS 460 (4) CS Elective (3) CS 470 (3)

1.5.2 Alignment of Curriculum with PLOs

Table 1.8 shows the alignment of program learning outcomes with the CS major core and electives. For convenience, the PLOs are repeated from Table 1.1, below:

1. Apply software design and engineering principles to develop and evaluate a computing-based solution to meet a given need.
2. Apply theoretical foundations, algorithmic principles, and computer organization fundamentals to analyze the tradeoffs involved in designing computer-based solutions.
3. Select appropriate tools and techniques for a given computing task, and quickly develop proficiency with new tools.
4. Collaborate and communicate effectively with others to accomplish professional goals.

5. Drawing on the foundations of a strong liberal arts education, make informed ethical judgments grounded in social and professional responsibility.

We recognize that some PLOs are not demonstrated anywhere at the mastery-level in our program (SSU-5), and no course provides the opportunity to demonstrate mastery of all PLOs. We recognize this as an assessment challenge the department must face (see Section 2 for discussion).

Table 1.8 Alignment of CS Courses with PLOs

Key	SLO introduced/reinforced at an introductory-level	SLO developed at an intermediate-level	SLO developed at an advanced / mastery-level

	B.S. in Computer Science PLOs				
	SSU-1	**SSU-2**	**SSU-3**	**SSU-4**	**SSU-5**
REQUIRED LOWER-DIVISION CORE					
MATH 161 Calculus					
CS 115 Programming I					
CS 210 Introduction to Unix					
CS 215 Programming II					
CS 242 Discrete Structures					
CS 252 Computer Organization					
REQUIRED UPPER-DIVISION CORE					
CS 315 Data Structures					
CS 351 Computer Architecture					
CS 355 Database Mgmt. System Design					
CS 370 Software Design & Dev.					
CS 415 Analysis of Algorithms					
CS 450 Operating Systems					
CS 454 Theory of Computation					
CS 460 Programming Languages					
CS 470 Adv Software Design Project					
COMPUTER SCIENCE ELECTIVES					
CS 340 Computer Security & Malware					
CS 349 Problem solving in a team env.					
CS 360 Object-oriented Programming					
CS 365 Computer Networking & Internet					
CS 375 Computer Graphics					
CS 390 Computer Science Colloquium					
CS 391 Computing Professions					
CS 425 Parallel Computing					
CS 452 Compiler Design & Construction					
CS 465 Data Communications					
CS 479 Computer Vision					
CS 480 Artificial Intelligence					

1.6 Interaction with Other Programs

We believe general education classes about technology literacy and computational thinking have strong value to undergraduate students. Over the period of this review, the Computer Science department regularly offered two GE courses: Introduction to Computers and Computing (CS 101, 3 units, area B3), and Programming I (CS 115, 4 units, area B3). Recently, as a response to the reform of the general education program at Sonoma State in 2019, CS 101 became “Computing Technology and You” in GE area E (life-long learning) and CS 115 stopped being a GE class. These changes are largely a response to capacity and staffing, rather than principles.

The CS program also serves related majors. Our first programming course (CS 115) is a required course in the electrical engineering major. It is also an option for a supporting course in the Environmental Systems B.A. program. Students value combining CS programs with several others on campus, and double-majors or CS-minors are popular among: programs in the Math & Stats department (applied statistics, mathematics with a bi-disciplinary concentration), programs in the Geography, Environment, and Planning department, and the Electrical Engineering major.

1.7 Reflections on Prior Program Review

The program challenges enumerated in the prior program review (see Appendix A) included:

- There is a lack of evidence that support and resources are sufficient to provide assurance that the program will retain its strength. Warned that “resource situation is precarious, with a danger of personnel burnout.”
- Close interaction between students and faculty in formal classroom settings was highlighted as a program advantage, but the lack of proximity between faculty offices and student laboratories negatively impacted informal interactions between faculty and students; recommended labs and capstone space on the first floor near faculty offices.
- We were urged to add Computer Networking as a required course rather than an elective and more software design topics.
- Recommended department and university coordinate on the development of general education courses that address students’ technological literacy and the relationship between technology and society.

These recommendations should be considered in light of the program having experienced personnel resignations possibly due, in part, to burnout (see Section 3.1); the fracturing of CS space across buildings has, if anything, reduced opportunities for informal interaction between students and faculty (see Section 4.4); the worsening of student-to-faculty ratio puts into jeopardy the close student-faculty interaction previously praised as a program strength (Sections 3.2 and 4.1).

1.8 Changes from Prior Program Review

The last major curriculum revision was in Fall 2007. That curriculum was aligned with the 2001 curricular guidelines for computer science published by the Association for Computing Machinery (ACM). The significant program changes since the previous program review were largely due to changes in general education at SSU and CSU graduation requirements (see Table 1.9 for a comparison of the current program with the program considered in the prior review).

Table 1.9. Comparison of B.S. CS degree requirements, since the last program review.

B.S. major in Fall 2022	B.S. major in Fall 2007
● 48 units of general education ● 49 units of major core requirements ● 9 units of CS upper-division electives ● 3 unit capstone requirement ● 10 units CS support courses ● 1 unit of general electives Total: 120 units	● 42–45 units of general education ● 49 units of major core requirements ● 9 units of CS upper-division electives ● 3 unit capstone requirement ● 10–12 units CS support courses ● 6–8 units of general electives Total: 124 units

In 2012, we formalized CS 496 as an option for students to work one-on-one with faculty on research-oriented capstone projects. We also expanded the offering of upper-division elective courses to include Computing Professions (CS 391), Game Programming (CS 330), Parallel Computing (CS 425) and Computer Vision Fundamentals (CS 497)—see Section 2.4 for related discussion.

1.9 National Trends in Computing Workforce and Education

1.9.1 Workforce Trends

The tech sector, which is the primary destination for computer science graduates, is infamous for its cyclical, boom-and-bust nature. California's economy, due to its outsized dependence on the tech industry, tracks the same boom-and-bust cycle [Levin23]. However, stepping back to see the larger picture shows decades of dramatic, sustained long-term growth. Data from the federal Bureau of Labor Statistics (BLS), as reported by the National Academies of Science, Engineering, and Medicine show that “employment in computer occupations grew by nearly a factor of 20 between 1975 and 2015, nearly twice as fast as production of [computer and information science] bachelor’s degrees.” BLS projected continued growth outpacing the overall job market through 2025, “particularly as computing becomes more central to a wider range of industrial sectors” [NASEM18].

While the long-term trajectory is one of overall growth, the shorter-term ups and downs can be dramatic. At the time of the previous program review, in 2007, the tech sector was still recovering from the turn-of-the-century dot-com bust. Fifteen years later, the largest tech companies are making news by laying off workers for the first time in years [Levin23, Lohr22]. An analysis in the New York Times from December 2022 provides broader perspective on this very recent shift:

> “*Still, overall employment in tech occupations has grown [in 2022], to a record 6.39 million in November, according to government statistics reported this month. That was slightly up from the previous month and a 12 percent increase from November 2021. Today, a majority of tech jobs are at companies outside the tech sector in industries like banking, retail, health care and manufacturing whose operations are increasingly becoming digital. These mainstream companies, unlike their Silicon Valley counterparts, did not go on manic hiring sprees during the pandemic. But they continue to invest in tech skills.”* [Lohr22]

Computing is ubiquitous, with its reach extending deeply and expanding in almost every economic sector, academic discipline, and aspect of modern living. The vast potential in computing, both as a job market and for facilitating numerous intellectual endeavors, will remain a catalyst for rising enrollments in undergraduate computer science, drawing interest from both majors and non-majors. While there may be variations in demand for computer science courses, it is expected that the demand will either persist at high levels or grow over the long run [NASEM18, finding 5].

1.9.2 Trends in CS Major Enrollment

Nationwide computer science enrollments have paralleled the larger economic trends. Figures 1.10 and 1.11 are drawn from slightly different data sets. Figure 1.10 shows bachelor’s degrees granted in the U.S. based on data from the National Center for Education Statistics from

1975–2012 and extrapolated from the Computing Research Association Taulbee Survey (which covers Ph.D.-granting institutions only) for 2013 and 2014. Figure 1.11 is drawn from the Taulbee data only—thus excluding institutions like SSU—and shows declared majors in computing fields (rather than graduates) from 1994–2018. The annual Taulbee Survey, despite the fact that it only covers research institutions, is the most systematically collected and extensive data set on CS undergraduate and graduate programs, and we will draw from it often in this document.

The biggest qualitative difference between the two graphs is the period since 2007, in which the number of CS majors more than quadrupled at Ph.D.-granting universities. The National Academies report notes that “the growth has also not been uniform across institutions. On average, institutions with very high research activity have experienced the greatest growth in CIS degree production among not-for-profit institutions between 2009 and 2015 (by 113 percent)...The average increases reported ranged from 75 percent in upper-level courses at non-doctoral CS institutions to 181 percent in upper-level courses at doctoral CS institutions, with similarly impressive increases in lower-level courses” [NASEM18].

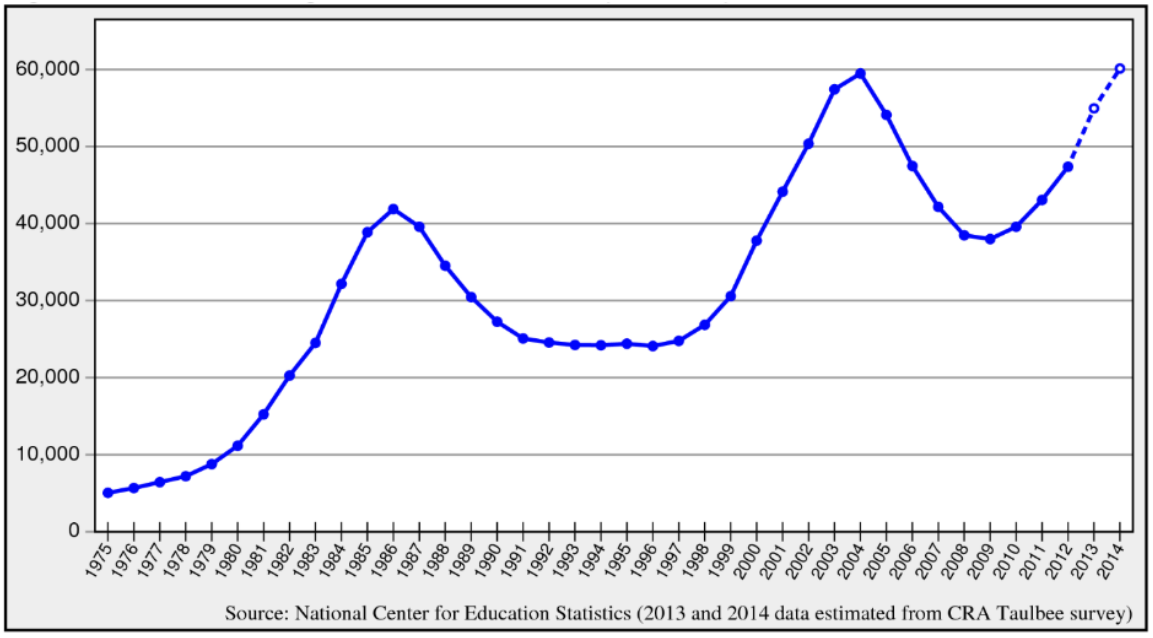

Figure 1.10. Computer science bachelor’s degrees granted in the United States (which lag enrollments by several years), 1975-2014, reproduced from [Roberts16].

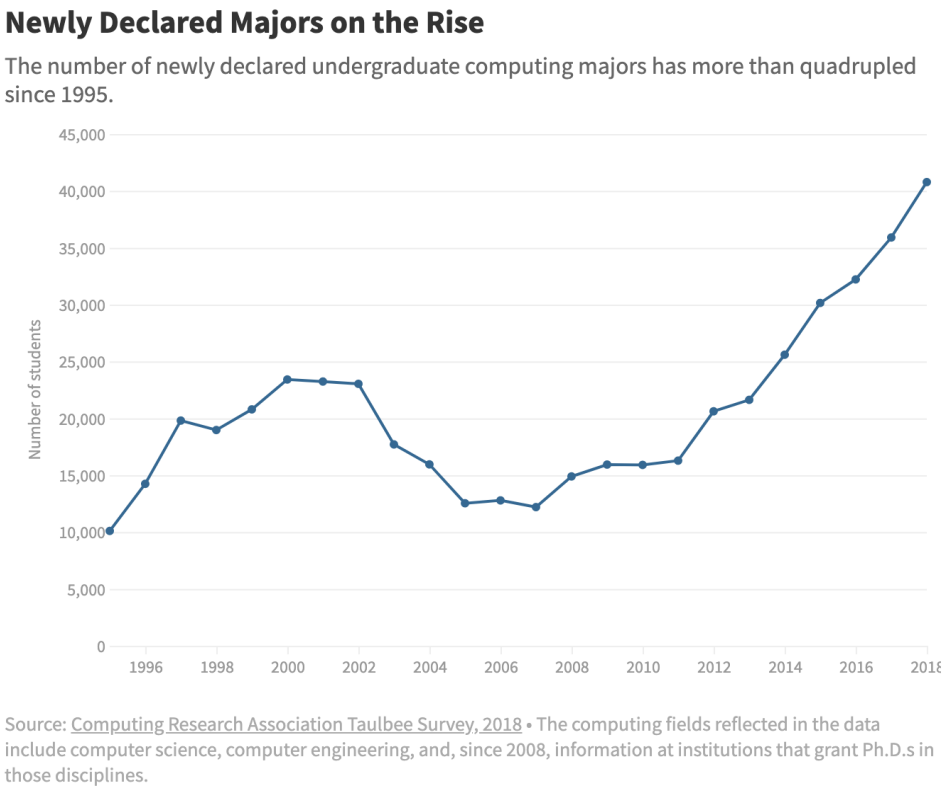

Figure 1.11 Newly declared computing majors at North American Ph.D.-granting universities, 1994-2018, reproduced from [Kafka20].

1.9.3 The “Capacity Crisis”

Computer science classrooms at colleges and universities throughout the United States are becoming increasingly overcrowded. This surge in enrollment is not only due to majors, but also non-majors who understand the significance of computing skills in today's economy. Unfortunately, this growth in enrollment is creating tremendous strain on computer science departments, as they have been unable to keep pace with the demand.

One of the main issues is that most new Ph.D.s in computer science opt to pursue industry careers, resulting in a shortage of individuals interested in educating the next generation. For example, while the number of undergraduates majoring in CS more than doubled from 2013 to 2017, the number of tenure-track faculty rose about 17% [Singer19]. Compared to traditional academic fields, where there are approximately five times more open faculty positions in computer science than there are candidates. Consequently, many institutions are unable to hire the necessary faculty and are forced to limit access to computer science courses and majors. Eric Roberts (CS professor, Stanford) reflects on this “Capacity Crisis” in CS education:

> *If one looks closely at the downturn of the 1980s, however, it quickly becomes clear that the reasons for the collapse in student enrollments had nothing at all to do with student interest. Student demand for computer science courses and degrees remained high*

> *throughout that period. Students in the mid 1980s did not decide against majoring in computer science but were instead prohibited from doing so by departments that lacked the resources to accommodate them.*
>
> *I believe that what happened in the 1980s is best described as a capacity collapse in which universities and colleges were simply unable to satisfy the growing level of student demand. Departments tried a number of strategies to increase their teaching capacity, including retraining faculty from other disciplines and hiring adjunct faculty from industry. In the end, however, demand overwhelmed capacity, and colleges and universities were forced to restrict admission to the computer science major, which gave rise to the subsequent downturn.* [Roberts16]

The strain on the capacity in CS has unintended social consequences. Singer observes “the shortage [of faculty] is creating an undergraduate divide of computing haves and have-nots — potentially narrowing a path for some minority and female students to an industry that has struggled with diversity” [Singer19]. Lack of seats in CS classrooms comes at the expense of students outside the major who want to build computer literacy and combine computational tools with other majors. This is consequential, as an important trend in the last 15 years has been the growing importance of computing to diverse academic disciplines and fields in the broader workforce. Therefore, it is no surprise that nonmajors want to engage with computer science in a variety of ways—from taking the occasional CS class to enrolling in new interdisciplinary majors (see Figure 1.12). At many leading institutions, a very large proportion of students are now choosing to take one or more CS courses, even though this is not required for many majors [Lazowska14]. About 95% of Stanford undergraduates take some CS course during their time at Stanford, and gender diversity within the major has followed suit [Liu19].

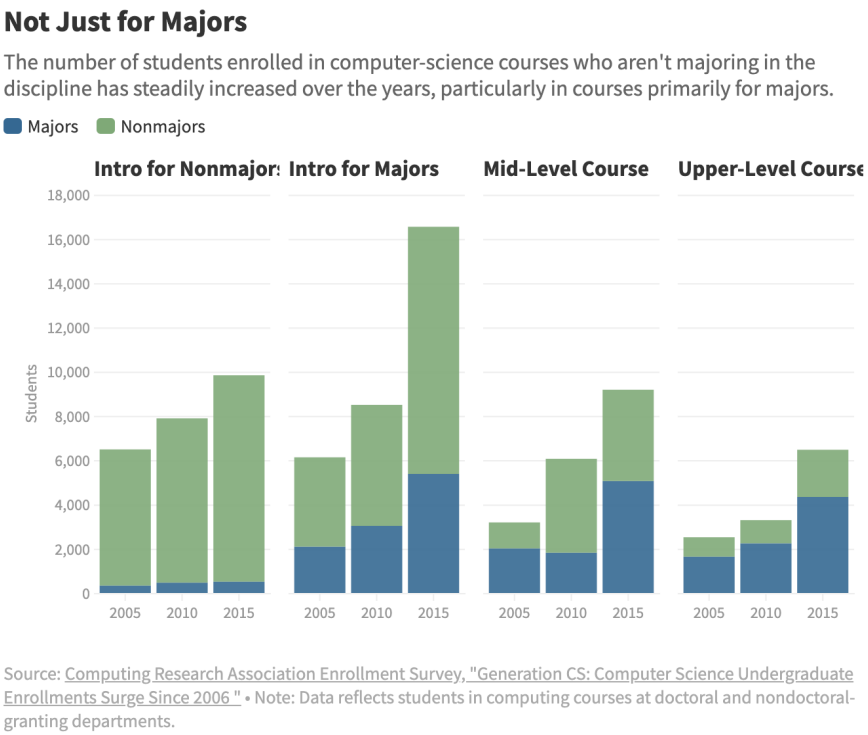

Figure 1.12 Trends in non-CS majors taking CS courses, reproduced from [Kafka20]

This is, of course, only for institutions fortunate enough to find strategies to offer those seats to non-majors. Mark Guzdial observes:

> *The greatest loss in the growing demand for CS classes isn't that there will be a narrower path for K-12 students to become professional software developers. As the [Computing Research Association's] Generation CS report showed, a big chunk of the demand for seats in CS courses is coming from CS minors and from non-CS majors. More and more people are discovering that Computer Science is useful, in whatever career they pursue. Those are the people who are losing out on seats. Maybe they first saw programming in K-12 and now want some more. That's the biggest cost of the capacity crisis. In the long run, increasing the computational literacy and sophistication across society could have even bigger impact than producing more professional programmers.*
>
> *Inability to meet the demand for seats in CS classes may limit the growth in our computing labor force. It may also limit the growth of computational scientists, engineers, journalists, and teachers —- in short, a computationally literate society.* [Guzdial19]

The well-funded, industry-supported efforts to get CS into every primary and secondary school in the US (through efforts like code.org) exist in conflict with the lack of capacity in higher education. When those young people want more CS in college, they find there are no seats available.

The capacity crisis threatens to exacerbate pre-existing crises of equity and inclusion in computer science. In a nutshell, CS shares a history of excluding certain ethnic groups with many STEM fields and has a history of excluding women that, sadly, is uniquely our own. It is not reasonable to assume that either of these conditions will naturally improve with time, as illustrated by Figure 1.13 and Figure 1.14.

Improving these numbers and reversing these dire trends are priorities for CS programs around the country, including ours. However, the capacity crisis threatens to worsen this situation. Eric Roberts recalls CS's original capacity crisis in the early 1980s:

> "*The imposition of GPA thresholds and other strategies to reduce enrollment led naturally to a change in how students perceived computer science. In the 1970s, students were welcomed eagerly into this new and exciting field. Around 1984, everything changed. Instead of welcoming students, departments began trying to push them away. Students got that message and concluded that they weren't wanted. Over the next few years, the idea that computer science was competitive and unwelcoming became widespread and started to have an impact even at institutions that had not imposed limitations on the major."* [Roberts16]

Figure 1.13 shows the disproportionate effect of this original 1984 capacity crisis on women's enrollments in computer science. We strongly concur with the National Academies of Science, Engineering, and Medicine's 2018 report on the CS capacity crisis that "[t]here is no guarantee that the representation of women and underrepresented minorities in CS will improve without a focused effort. Retention is always a challenge, and adverse conditions associated with high demand for courses—as well as actions taken by institutions in order to manage enrollments—could negatively impact the inclusiveness of undergraduate computing programs" [NASEM18, finding 7]. Largely because of these concerns, we have resisted impaction despite immense pressure on resources over the period of this program review.

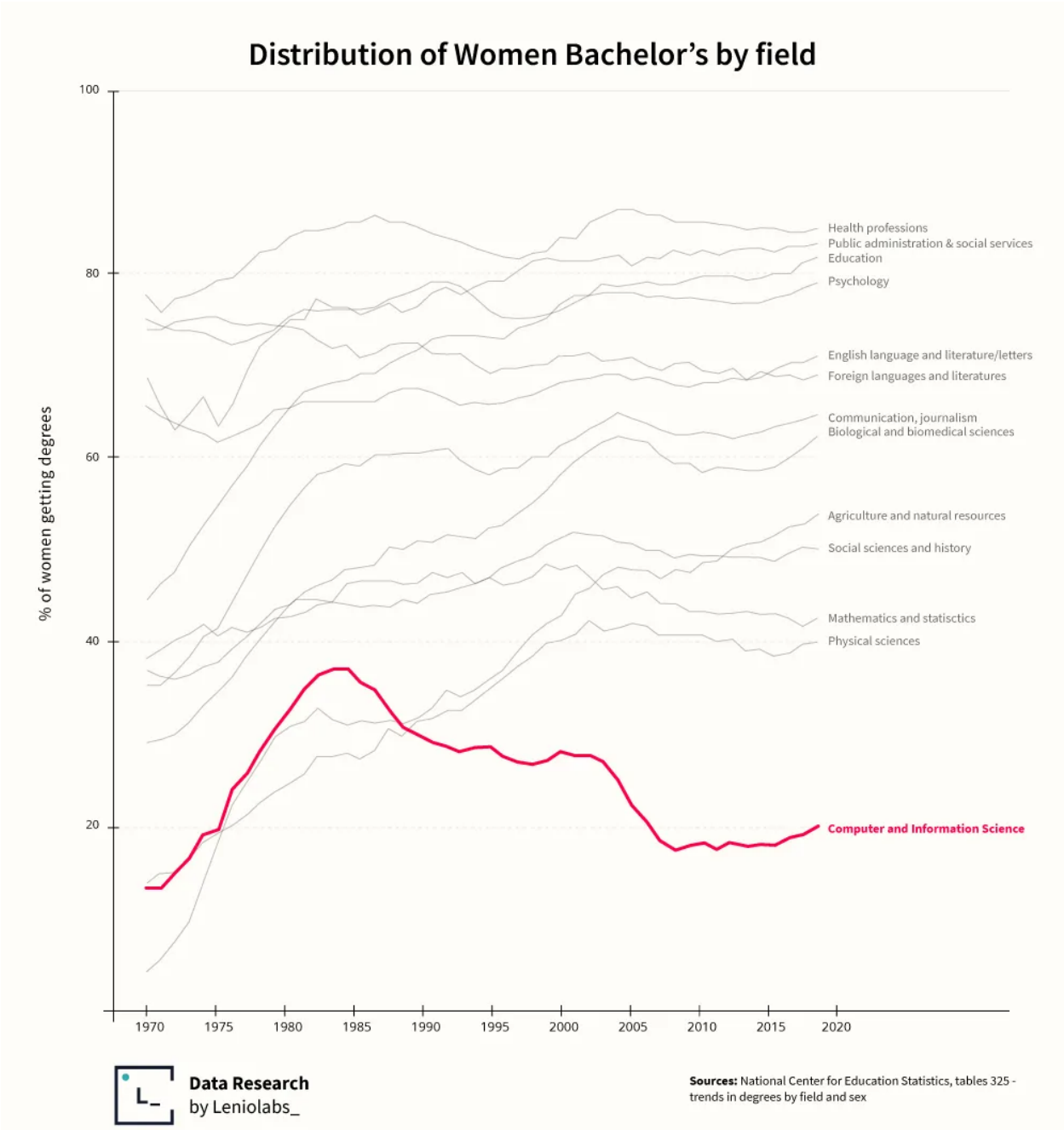

Figure 1.13 Distribution of female students' bachelor's degrees by field, highlighting computer and information science, reproduced from [Schvartzman].

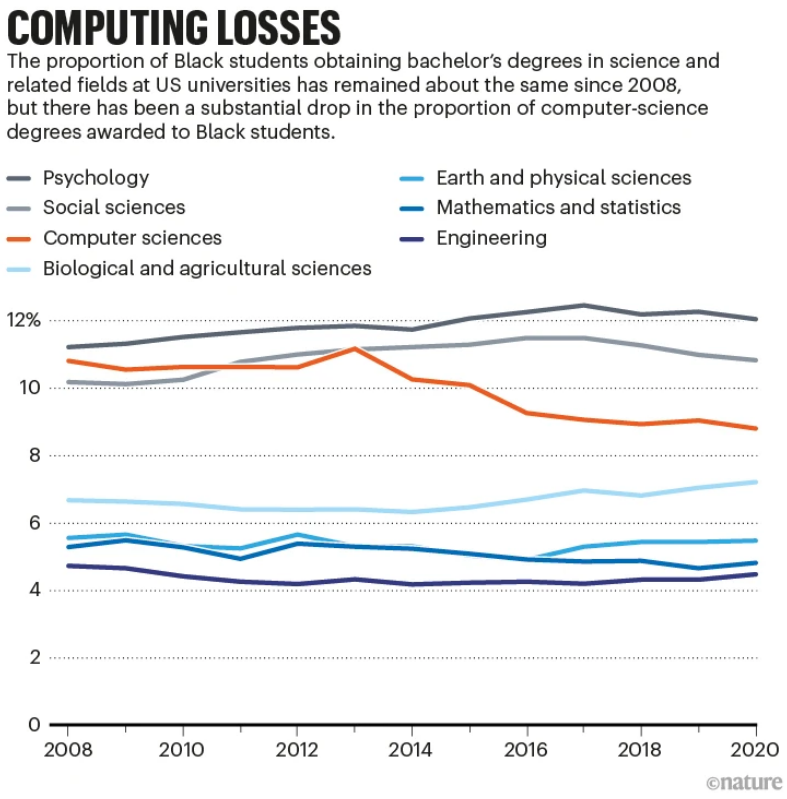

Figure 1.14 Distribution of Black students' bachelor's degrees by field. Computer Science, in red, is low and trending downward. [reproduced from Newsome22]

1.9.4 Faculty Recruitment and Diversity Challenges

Because of the robust industry demand for CS graduates, faculty jobs are not considered as the ultimate prize nor even particularly attractive. In the most recent survey of North American Ph.D.-granting computing departments, fewer than 15% of graduates took positions in academia, and only 4.2% at non-research institutions [Zweben22]. Interest in all types of academic positions has dropped over time (see Figure 1.15). Consider, for example, this anecdote from the experience of the CS department of UT Austin:

> *"The demand is unbounded," said Don Fussell, chairman of the [UT-Austin] computer science department. The university is looking to hire several tenure-track faculty members in computing this year, he said, but competition for top candidates is fierce. "I know of major departments that interviewed 40 candidates, and I don't think they hired anybody." [Singer19]*

Unsurprisingly, a lot of searches in CS fail. Across all North American institutions for positions starting in Fall 2022, 55% hired fewer candidates than they were searching for, which is worse than the pre-pandemic numbers [WillsCRA22, WillsTR22].
The problem is particularly acute for teaching institutions for several reasons. Salary and prestige are obviously two. Another key reason is the high percentage of nonresident Ph.D.

students, who tend to be aware only of the research institution piece of the US higher education landscape. Furthermore, Ph.D. programs—quite reasonably given stats above—don't prepare CS students for positions with high teaching loads. It is a huge ask for foreign Ph.D. students to develop the linguistic and cultural competency to succeed at non-research institutions during their Ph.D. years. Thus, nonresident aliens are underrepresented in faculty positions: 65.3% of CS Ph.D. students, but 16.3% of assistant professors at research institutions and 6.4% of assistant professors at non-research institutions [Zweben22].

The National Academies of Science, Engineering, and Medicine's 2018 report on the CS capacity crisis cautions:

> D*epartments facing sharp increases in demand for computing courses have experienced significant strain on a wide range of resources. Failure to respond thoughtfully to the demand and the resource deficits will result in adverse conditions for students, faculty, the programs, and the institution as a whole in the near or long term. Conditions such as an unwelcoming academic climate and loss of faculty members can be especially harmful in the long term.* [NASEM18, finding 8].

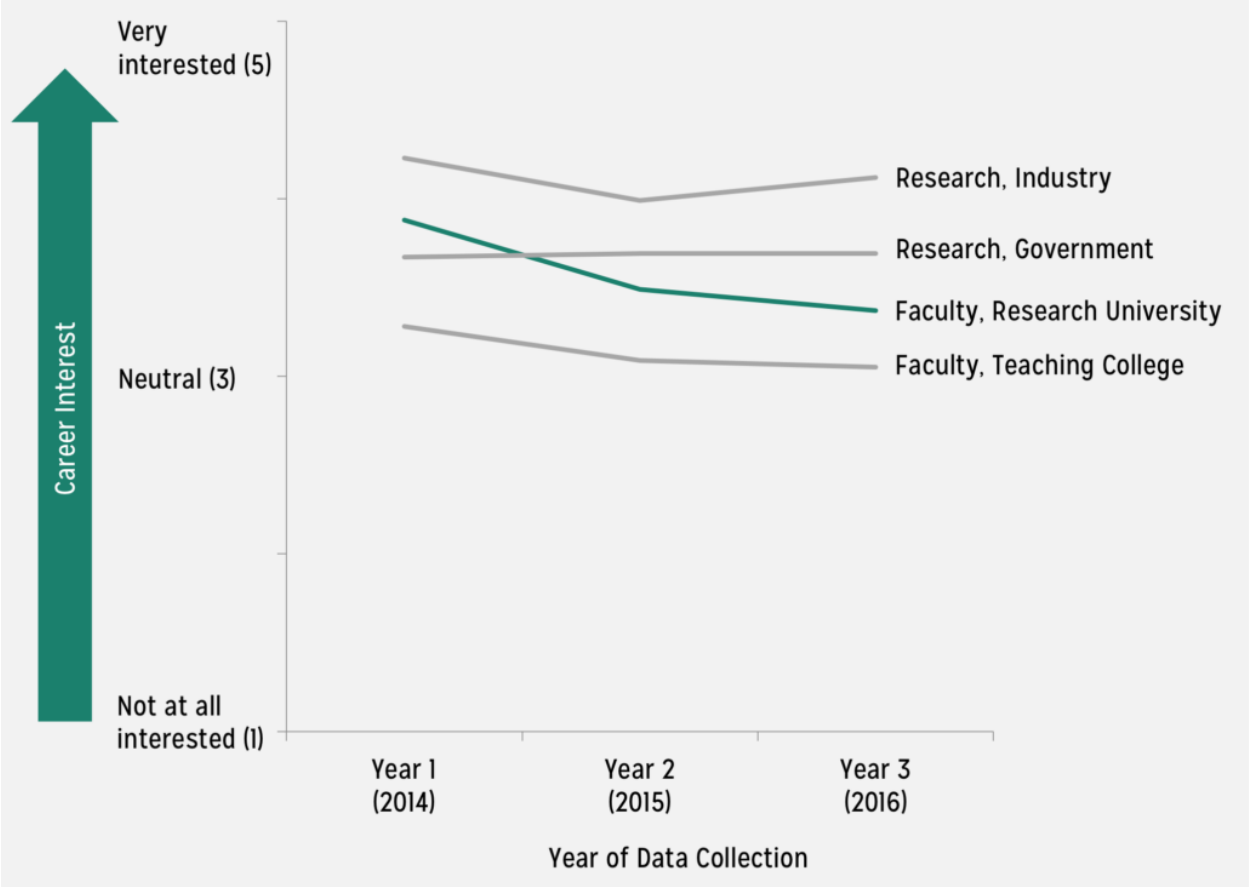

Figure 1.15 Doctoral students' interest in different types of academic careers, reproduced from [Tamer18]

2. Assessment

With the exception of individual course evaluations (SETEs), we are not systematically or quantitatively gathering student or alumni feedback about our program. However, we do have numerous informal ways of staying in touch with our alumni:

- LinkedIn alumni group with 450+ members
- Inviting alumni speakers to the Computer Science Colloquium (CS 390)
- Inviting alumni to participate in panels about their careers and job searches for Computing Professions (CS 391)
- Alumni participation in events hosted by our student organizations, the Computer Science Club and the Women in Computer Science Club (WiCS), ranging from visiting SSU to discuss career topics to hosting company tours
- Inviting alumni to teach for us. Over the review period, more than a dozen of our alumni have taught courses for us. These courses are generally either lab sections of combined lecture-lab courses, or topics that are highly practical and benefit from an industry viewpoint. For example, in the current semester, we have alumni teaching CS 355 (Introduction to Database Management Systems), CS 375 (Computer Graphics), and CS 385 (Selected Topics in Computer Science).
- Alumni hiring pipelines at local companies. Notable examples include Keysight in Santa Rosa, FIS Mobile in Larkspur and San Francisco, Disney Streaming in Point Richmond, and Broadcom Inc. in Petaluma.

These alumni provide valuable career connections for our students, and they also keep our faculty informed about the ever-evolving computing industry.

2.1 Direct Assessment and Assessment of PLOs

The Department of Computer Science uses an evolving set of tools for program assessment. These include traditional exams and quizzes, programming projects, lab assignments, and student presentations based on their senior capstone projects or research activities.

For convenience, the PLOs are repeated from Table 1.1, below:

1. Apply software design and engineering principles to develop and evaluate a computing-based solution to meet a given need.
2. Apply theoretical foundations, algorithmic principles, and computer organization fundamentals to analyze the tradeoffs involved in designing computer-based solutions.
3. Select appropriate tools and techniques for a given computing task, and quickly develop proficiency with new tools.
4. Collaborate and communicate effectively with others to accomplish professional goals.
5. Drawing on the foundations of a strong liberal arts education, make informed ethical judgments grounded in social and professional responsibility.

The following is a broad categorization of Computer Science courses and their educational contribution to a degree in this discipline.

Theoretical courses.

Courses including Discrete Structures (CS 242), Analysis of Algorithms (CS 415), Theory of Computation (CS 454) form the theoretical backbone of a computer science education. Though these include programming projects, the learning emphasis is on those theories upon which the science of computing rests. The curriculum for these is based on national curriculum standards, and the assessment of student learning is generally based on quizzes and exams.

These courses directly address PLO SSU-2 and, to some extent, PLO SSU-3.

Foundational Programming Courses.

These courses (CS 115, CS 215, CS 252, and CS 315) consist of a combination of lecture and closed labs. They introduce students to different programming languages and paradigms and progressively employ theoretical and practical problem-solving methods to develop programming projects. Each of these courses meets once a week in a closed lab for the duration of three hours. Each lab has its own objectives and deliverables, which are generally based on the on-going discussion in the lecture. They are designed to focus students' attention on either a particular aspect of programming or the applications of data structures and algorithms that benefit from their use. The main objective of the lab is learning and discovery through individual or small-group activities and close contact with the instructor.

In addition to lab assignments, in each of these courses students are assigned multiple programming projects during the semester, which they are expected to complete on their own. These programming projects synthesize concepts that students learn in the lecture and labs. Projects often require a large amount of time to comprehend and complete.

Exams and quizzes are used to assess student learning of the material presented in the lectures, labs, and programming projects. Feedback on labs and projects, sometime in a one-on-one setting, help students to learn and employ better programming practices.

These courses address mainly PLO SSU-1, but also SSU-3 and SSU-4.

Software Engineering Courses.

The concepts of Software Engineering run through many programming courses. However, Software Engineering (CS 370) and Advance Software Design Project (CS 470) -- which serves as a capstone course -- more formally introduce students to the theory and practice of software design and engineering. In these courses, students work in groups and projects are generally more extensive and explorative.

Each of these courses has a semester project where students spend 6 to 8 weeks, in groups ranging from 3 to 5, developing a large computer project. Almost always, each group proposes a project that the team members conceive and if approved, they go through the stages of software development to design and implement a working product. As they go through this exercise, each team, sometimes on a weekly basis, presents their work to the class for analysis and feedback.

The themes that we have employed in these courses include Android and iOS, and Full-stack, cross-platform, application development.

We have used different venues for students in these courses to present their final projects. These include the demos in the lobby of the Darwin Hall, poster presentations at the Science Symposium, and live and/or video presentations to CS students during one of the Colloquium (CS 390) class meetings, which on the average, enrolls 60 CS majors.

Assessment in these courses is based on the novelty of the project proposal, team work, and the implementation of the proposal.

These courses address PLOs SSU-3 and SSU-4 and, to a lesser extent, SSU-1 and SSU-5.

400-level courses.

These four Core courses cover the traditional advanced topics in Computer Science. Analysis of Algorithms (CS 415) and Theory of Computation (CS 454) are heavily theoretical, Operating Systems provides an in-depth study of the Software Architecture of operating systems building blocks and how they tie together, and Programming Languages (CS 460) is a combination of the study of programming language paradigms and the application of Theory of Computation in crafting compilers and interpreters.

Assessment in CS 415 and CS 454 is based on quizzes, exams and projects, with the emphasis on quizzes and exams. In comparison, projects play a larger role in the assessment in CS 450 and CS 460.

These courses address PLOs SSU-1, SSU-2, and SSU-3.

Hardware-Learning courses.

Introduction to Computer Organization (CS 252) and Computer Architecture (CS 351) introduce students to the interaction between software and hardware and play an important role in students' understanding of the layers through which programs travel when their instructions are being executed.

These courses address SLOs SSU-2 & SSU-3.

Assessment in these courses are through quizzes, homework assignments, quizzes exams, and programming projects.

Computer Science Electives.

Each student is required to take 9 units of CS Electives, a collection of courses that introduce students to application areas of Computer Science. Whenever we have solicited students' feedback about our program, this particular category has come up because it is just about the only place in our curriculum where students experience the practical applications of what they learn in the Core courses. On the other hand, due to staffing issues, we have hardly been able to offer many of the courses in this category on a reliable basis. In fact, there are several courses in this category that we have not offered during this review cycle. We intend to remove these courses and when we redesign our curriculum, add courses to this category and take a more realistic approach to adding courses to this category.

Many courses in this category include projects and assessment is done through exams, quizzes, and review of projects.

The PLOs that the courses in this category address differ from course to course.

Capstone experience and direct-study courses.

All students are required to take one of the Advance Software Design Project (CS 470) and Senior Research Project (CS 496) as their senior capstone experience. As mentioned above, CS 470 is a software engineering course. CS 496 is a directed, one-on-one, research contract course that can be exclusively theoretical or it can be a major programming application that is the product of research in a particular area of computer science. Another course that is usually used as a precursor to CS 496 is the directed-study, one-on-one, Special Studies (CS 495), which can be applied to the CS Electives requirement of the Computer Science degree. Several times the involvement of students in CS 495 and CS 496 has resulted in publications and conference presentations.

The PLOs that each address vary depending on the particular topic.

Assessment in CS 495 and CS 496 take place usually on a weekly basis through a one-on-one meeting with the lead instructor. Students usually present their work either in Colloquium or in the form of poster presentations at the Science Symposium.

2.2 Indirect Assessment Findings

We have taken the snapshot of students' sentiment through the use of exit surveys, focus groups, and alumni surveys. Additionally, we are in close contact with a sizable group of our

alumni who work in the local industries that have national and international presence. Notable examples include Keysight in Santa Rosa, FIS Mobile in Larkspur and San Francisco, Disney Streaming in Point Richmond, and Broadcom Inc. in Petaluma.

2.2.1 Exit Survey, AY 2015

In an exit survey in 2015 of ~36 students, using a scale of 1 (least favorable) to 5 (most favorable), students rated a variety of aspects of the program. They rated faculty's knowledge of subject matter positively (4.3). They appear generally happy with the physical lab rooms (4.04), hardware (4.33), and hours of operations (4.17). They rated advising availability (4.3) and quality (4.24) in positive terms. They rated availability of electives (3.14) and their ability to enroll in courses in general (3.53) lower. They showed opportunities for improvement with respect to student club activities (3.5).

The following is a selection of responses to the question "What are the best attributes of the CS program?" The number of students whose comments echoed similar themes are noted in parentheses for each response.

- Instructors are knowledgeable, approachable, available outside classroom, good advisors, range of skills (18)
- Small class sizes (9)
- Peer and classmates (5)
- Computer labs (4)
- Referral to jobs (3)
- Up-to-date topics, engaging material (3)
- Hands-on projects (3)

Their suggestions for areas that need improvement generally fall into the following categories:

- More professors
- More Object-Oriented programming
- Course availability
- More Electives
- Resume review and mock interviews

The following is a selected list of comments that students offered.

1. I have really enjoyed finishing my CS education at SSU. Students and faculty seem to all know each other and form a tight-knit group, which I was able to notice from day 1.
2. Thank you for the overwhelming positive experience in Computer Science
3. Programming Languages and Senior Capstone have improved my coding skills drastically.
4. Overall very happy (5)

5. High quality of teachers and courses. Projects went above and beyond for preparation for a job environment.
6. I truly get the sense that the professors genuinely care about my personal life as much as they do my education

2.2.1 Exit Interview, Spring 2022

In Spring 2022, Dr. Martha Shott (SSU, Department of Mathematics and Statistics) conducted an exit interview of ~25 students who were expected to graduate that semester. This group consisted of students taking CS 470 and CS 496 that semester. The following are some excerpts from this interview. Dr. Shott's report is included, in full, in Appendix B.

Strengths of the program.

- Faculty expertise and their connection with the industry.
- Computer lab key-card access.
- Lab sections in lower-division courses. Not only do they find them helpful in their learning of the material, they also provide an environment for students to bond together and learn from each other.
- In CS 470, they value group work, their ability to choose their group members, propose their semester projects, and implement an extensive project. In general, they enjoy developing semester-long projects.
- They find CS 215, CS 315, and CS 460 specifically to contribute significantly to their education. CS 460 almost always includes a semester-long project.
- Campus support resources – students appreciated the free tutoring that LARC provides for lower-division CS courses.

Areas for departmental consideration.

- Students like to see CS 470 to become a two-semester course.
- More clear communication regarding lab access via key card. Some students were not aware they could have that level of access.
- Limited Elective offerings, unpredictable schedules, and not enough seats.
- Quicker feedback on assignment.
- Students would like to see courses such as Linear Algebra[1], iOS programming, and Game Programming[2] to be offered more frequently.

Areas of concern.

- Students noted that the loss of 4 faculty was felt across the department.
- Some noted the loss of instructional time due to fires, power outages, etc. and how this is making them feel unprepared for their careers after graduation.

[1] Linear Algebra is offered by the Mathematics & Statistics Department and counts as a Support Course for CS Majors. The contents of the course help students in other CS courses. In matters of this nature, we have found the Mathematics & Statistics Department to be supportive of our program.
[2] iOS Programming & Game Programming are CS Electives.

2.2.2 Alumni Survey, Fall 2022

In Fall 2022, we invited the members of our LinkedIn alumni group to participate in an anonymous survey about their lives after graduation and their feedback about our program. The results of this survey are detailed in Section 5.4 of this report, under the topic of Student Success. The result of the survey demonstrates that our alumni are satisfied with the major (4.3 out of 5), rate the intellectual challenge that they received in our program high (4.15), rate the professional competency of the faculty high (4.35), as they do the quality the instruction they received (4.21), support and concern for their academic success (4.25), support and concern for them as individuals (4.35), and just in other surveys, they are not as satisfied with the frequency and the variety of the electives that we offer. We provide more commentary on this in Section 5.4.

2.3 Assessment Reflection

As we mentioned above, CS Electives introduce students to application areas of computer science. Students' frustrations with the frequency with which we offer Electives, the variety of Electives that we offer, the absence of a long-term schedule that would help them plan to take the Electives that they prefer, and the limited number of sections that we offer is understandable. The lack of a predictable course rotation is due to staffing issues that the department has perennially endured. The scheduling priorities are to staff the CS major Core courses first, and then decide which electives to offer based on the capacity of remaining faculty or availability of qualified adjunct faculty.

Over time, we have found effective ways to communicate with our majors. We have drastically improved our web presence and frequently share internship and professional events with our students via email listserv. Based on students' feedback, we will more clearly communicate benefits such as the ability to access our labs during off hours via key card.

We deeply share their concern regarding the loss of our four tenure-track faculty members and the loss of educational opportunities due to fires and the pandemic.

2.4 Assessment-Driven Curricular Changes

Based on prior student feedback and our own perception of new application areas of computer science, over the period of this review cycle, we added several electives to our curriculum to be responsive to student need:

1. Computing Professions (CS 391, 1 unit). This course provides a bridge between students' computer science education and the professional world of computing. They learn how to write a resume, cover letter, and how to communicate effectively as professionals during the interviews. Our alumni are frequently invited to give presentations in this course and to help students with networking.

2. Game Programming (CS 330, 3 units). This course is an introduction to the theory and practice of video game design and programming. Video games combine, in real-time, concepts in computer graphics, human-computer interaction, networking, artificial intelligence, computer aided instruction, computer architecture, and databases. This course introduces students to a variety of game engines and frameworks and explores artificially intelligent agents. Students will work as part of a team to create a complete description document for a computer game and implement a prototype of the game.

3. Parallel Programming (CS 425, 3 units). This course provides an overview of parallel patterns, programming models, and hardware. Topics include parallel performance analysis; types of parallelism; parallel decomposition of tasks; shared vs. distributed memory; synchronization; hands-on experience with multiple parallel programming models; and architectural support for parallelism.

4. Computer Vision Fundamentals (CS 479, 3 units). This course covers algorithms for face detection and face recognition are now widely employed for surveillance, security and entertainment applications. This course will delve into the study and implementation of such algorithms for detecting generic objects (pedestrians, animals, buildings, traffic signs, etc.). It will involve learning about (i) image filtering operations such as smoothing, thresholding and edge detection, (ii) interest point detection and representation methods such as SIFT and HOG, and (iii) machine learning classification techniques such as SVM and convolutional neural networks.

In addition, we have changed technologies and methods that we use in a number of our courses. Examples of this type of change are:

- Change of the web-based paradigm in Database Management System Design (CS 355) from Perl to Django (Perl-based) and then to NodeJS (JavaScript-based.)
- Software Engineering has gone through many changes in recent years. In response to that, not only the topics that are covered in CS 370 changed, we also have introduced development environments such as Android Application Development and Alexa Development platforms.
- Programming Languages (CS 460) has recently become more functional programming language-centric.

2.5 Future Assessment Plans

Our intention is to redesign our curriculum in the near future (see Section 6, Reflection and Plan of Action). Any such redesign will incorporate assessment that crosses the boundaries of courses, to prevent course-level changes from degrading, in aggregate, program rigor as it is so valued by both students and employers. Certainly, assessment of capstone work is desirable as a summary artifact of their CS experience. Assessment of the program beyond the capstone is also desired, but is complex due to the multiple paths through the program by FTF and transfer students. As such, assessment of CS315 (either or both pre- and post-assessment) and an exit

assessment covering 400-level topics are reasonable places for assessment in our current curriculum. In the prior review period, we had previously employed the ETS Major Field Tests in Computer Science as a type of program-level assessment exam. This practice was suspected due to the logistical complexity of paying for student testing fees due to policy (and not due to lack of funding).

3. Faculty

3.1 Composition

During this review cycle, the Department of Computer Science went through a significant number of changes in its faculty membership. We started in Spring 2009 with 5 tenured faculty and one tenure-track faculty.

Table 3.1. Summary of CS faculty composition, Fall 2008

Faculty Member	Yr. Hired	Degree
Ali Kooshesh	2002	PhD, Computer Science, University of New Mexico, 1992
George Ledin	1984	JD, University of San Francisco, 1982
B. Ravikumar	2001	PhD, Computer Science, University of Minnesota, 1987
Suzanne Rivoire	2008	PhD, Electrical Engineering, Stanford University, 2008
Lynn Stauffer	1994	PhD, Computer Science, University of California, Irvine, 1994
Tia (Marcia) Watts	2001	PhD, Computer Science, University of Pittsburgh, 1997

Over the period since the last review, we hired 6 new tenure-track faculty; an existing tenured faculty member accepted the position of Dean of the School and later retired; 2 faculty members entered the FERP program and subsequently retired; and 4 of our 6 new hires left the University. During the 2021-2022 academic year, in the span of 7 months, the department lost all four of its tenure-track faculty.

Table 3.2. Summary of CS faculty composition, Fall 2022

Faculty Member	Yr. Hired	Degree
Gurman Gill	2015	PhD, Electrical Engineering, McGill University, 2009
Mark Gondree	2016	PhD, Computer Science, University of California, Davis 2009
Ali Kooshesh	2002	PhD, Computer Science, University of New Mexico, 1992
B. Ravikumar	2001	PhD, Computer Science, University of Minnesota, 1987
Suzanne Rivoire	2008	PhD, Electrical Engineering, Stanford University, 2008

Dr. Lynn Stauffer became the Dean of the School of Science and Technology in Fall 2010, returned to the Department in Spring 2021 as a half-time professor, and retired in Fall 2021 taking advantage of the Voluntary Separation Incentive Program (VSIP). Dr. Rivoire was tenured

and promoted to Associate Professor in Fall 2014 and to Professor in Fall 2019. Dr. George Ledin, Jr., entered the FERP program in Fall 2015, but due to health issues, terminated his position in Spring 2018 and subsequently passed away in February 2022. Drs. Gurman Gill and Mark Gondree were tenured and promoted to Associate Professor in Fall 2020. Dr. Tia Watts, entered FERP in 2018 and finished her service in Fall 2022.

The following two tables summarize these events. No exit interviews were conducted by Faculty Affairs or Human Resources for any CS faculty resignations. Upon the request of the CS department, a series of informal exit interviews surveys were conducted in Spring 2022 by Dr. Paolucci Callahan in his position as Interim Associate Vice President of Faculty Success. These suggested the resignations may have been related to dissatisfaction with aspects of the career at Sonoma State, to mistreatment by students and administration/staff outside of the CS department, and unrelated to the CS department or its faculty.

Table 3.3. Summary of failed searches, hires, and resignations, Spring 2009–Spring 2022

AY 2013–14	Failed Search
Fall 2015	Hired Dr. Gurman Gill as an Assistant Professor
Fall 2015	Failed Search
Fall 2016	Hired Dr. Gondree as an Assistant Professor
Fall 2017	Hired Dr. Anamary Leal as an Assistant Professor
Fall 2018	Hired Dr. Shubbhi Taneja as an Assistant Professor
Fall 2020	Hired Drs. Nina Marhamati and Sabidur Rahman
Fall 2021	Drs. Leal and Taneja resigned their positions
Spring 2022	Drs. Marahamati and Rahman resigned their positions

Table 3.4. Summary of retirement, promotion and tenure, Spring 2009–Spring 2022

Fall 2010	Dr. Lynn Stauffer became the Dean of the School of Science and Technology
Fall 2014	Dr. Rivoire was tenured and promoted to the rank of Associate Professor
Spring 2018	Dr. Ledin terminated FERP early, due to health reasons
Fall 2019	Dr. Rivoire was promoted to the rank of Professor
Fall 2020	Drs. Gurman Gill and Mark Gondree were tenured and promoted to the rank of Associate Professor

Fall 2021	Dr. Stauffer returned to the CS department, but was on leave that semester
Spring 2021	Dr. Stauffer returned with 50% workload
Fall 2021	Dr. Stauffer retired early, via an early exit program
Fall 2022	Dr. Watts retired at the end of her FERP, as planned

Figure 3.5 shows the numbers and change in the tenured and tenure-track faculty during the review period. Figure 3.6 compares the number of tenured and tenure-track faculty in CS departments at peer CSUs as of Fall 2022. These counts exclude FERP or emeritus faculty. Given the number of majors we serve, the department at SSU appears to be one of the lowest resourced CS departments in the CSU system.

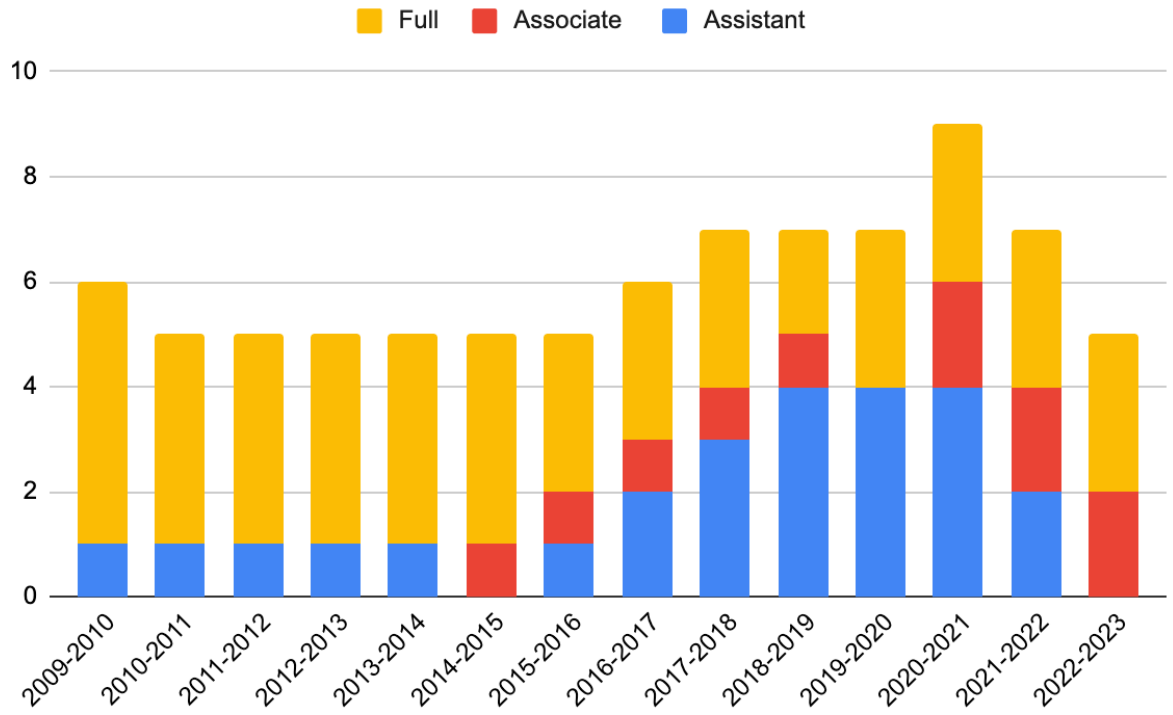

Figure 3.5 Changes in faculty composition by rank, Fall 2009–Fall 2022

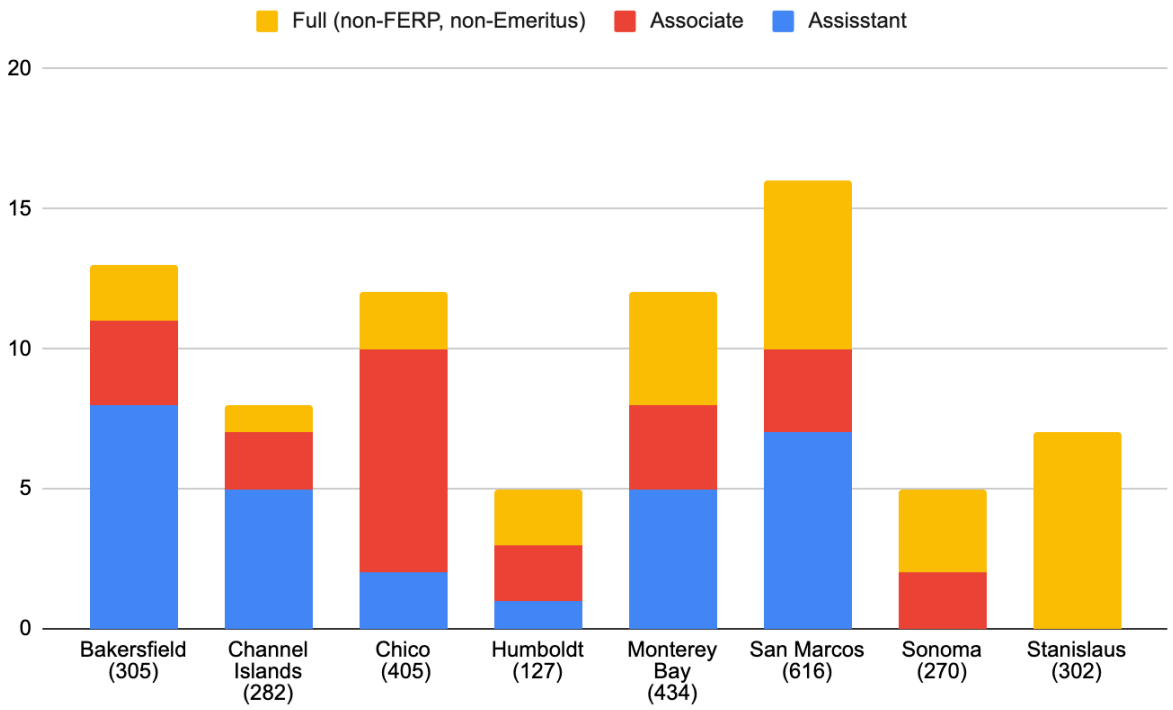

Figure 3.6. Faculty composition for CSU campuses with student populations comparable to SSU; x-axis shows the number of CS majors at campus

The department also employs “temporary” faculty, which includes Mr. Glenn Carter, the department’s only full-time lecturer. Mr. Carter has been a full-time lecturer with the department since 1997. His primary responsibility is CS 101, the department’s most regularly-offered GE course. A small pool of additional adjunct faculty are used strategically across the program. This pool is drawn from two main sources: SRJC faculty and program alumni who are now industry professionals.

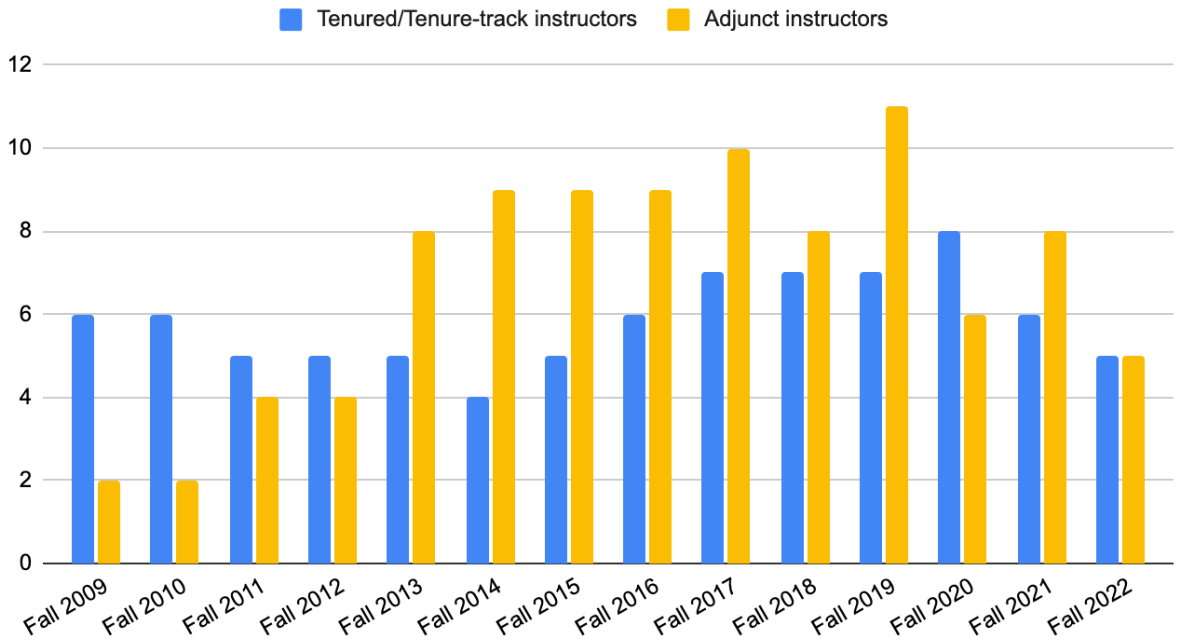

Figure 3.7. Comparison of permanent and temporary faculty, Fall 2009–Fall 2022

The retention and recruitment of adjunct faculty is a perennial challenge for the department. The most recent lecturer pool refresh occurred in Spring 2020, the outcome being zero growth in our lecturer pool. Compared to pools in other SSU departments, ours is composed mainly of local industry professionals and faculty with full-time employment elsewhere. In particular, while our adjuncts sometimes work enough for the department to be eligible for full-time lecturer positions, they have turned-down this offer and continued working as adjuncts (see Table 3.8 for a recent history of this for three unnamed adjuncts). Over the period since the last review, there has been only one instance of adjunct faculty earning entitlement and accepting work under that entitlement; that individual retired in Fall 2019.

Table 3.8. Data showing the history of adjuncts passing on eligibility for full-time positions

Eligible Year	Entitlement	After Eligibility	Employer
2019-2020	8 WTUs	Continued teaching until Spring 2020	Disney+

2017-2018	4 WTUs	Continued teaching until Fall 2020	Broadcom, Inc.
2020-2021	8 WTUs	Stopped teaching in Fall 2020	Sonoma Technology, Inc.

While we endeavor to use adjunct faculty strategically, teaching those classes where their prior industry experience is of most benefit to our students (such as in database management and software engineering topics). The amount of upper-division courses delivered by adjunct faculty whose highest degree is a B.S. in CS is of potential concern (see Figure 3.9). While have done no review of the role adjunct faculty play in the curriculum of other B.S. in CS programs, in other fields at SSU it does not seem common to employ adjunct faculty to teach a significant fraction of upper division coursework

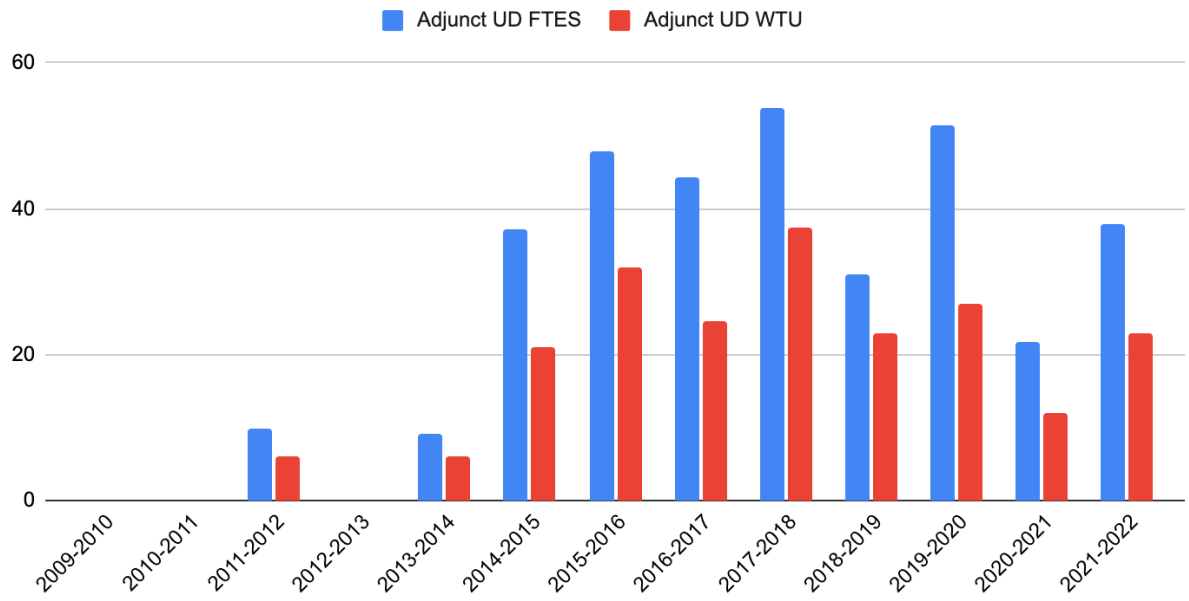

Figure 3.9. FTES and WTUs associated with adjunct faculty teaching upper-division CS courses, while holding a B.S. in CS as a terminal degree, 2009–2022

3.2 Faculty Workload

The metric of full-time equivalent students (FTES) is analogous to what fraction of a full-time student's 15-credit full-time load is delivered by a particular course, summed over the number of students enrolled in that course. For a particular semester, we can sum the FTES delivered across all courses in a department. Figure 3.10 shows the FTES delivered by the department in total, differentiating this as FTES delivered by permanent faculty and by adjunct faculty. Figure 3.11 shows the FTES when excluding the effects of our large-format general education course CS 101.

As Figure 3.11 demonstrates, starting in Fall 2014, on the average, close to half of the CS department's FTES, driven by courses that are required for a BS in CS, have been delivered by temporary faculty. With few exceptions, part-time faculty who teach courses that are required by the major have full-time jobs outside of SSU and, as a result, the department chair and the tenured/tenure-track faculty pick-up the burden of responding to students' needs.

The related metric of full-time equivalent faculty (FTEF) relates to how many full-time faculty used to teach the FTES. (This number excludes faculty time not used for teaching, such as department chair release time.) Figure 3.12 shows the average FTEF in the same terms as 3.10. Again, Figure 3.13 shows the FTEFs when excluding the effects of our large-format general education course CS 101.

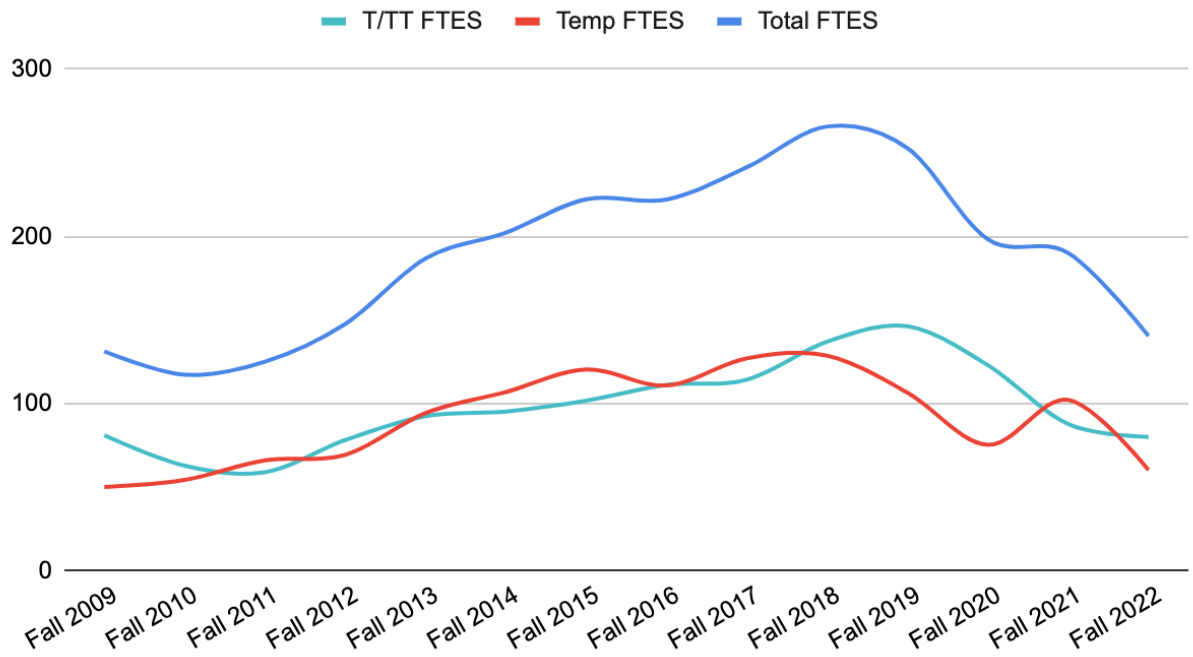

Figure 3.10. FTES for all CS classes (averaged across Fall and Spring), 2009–2022

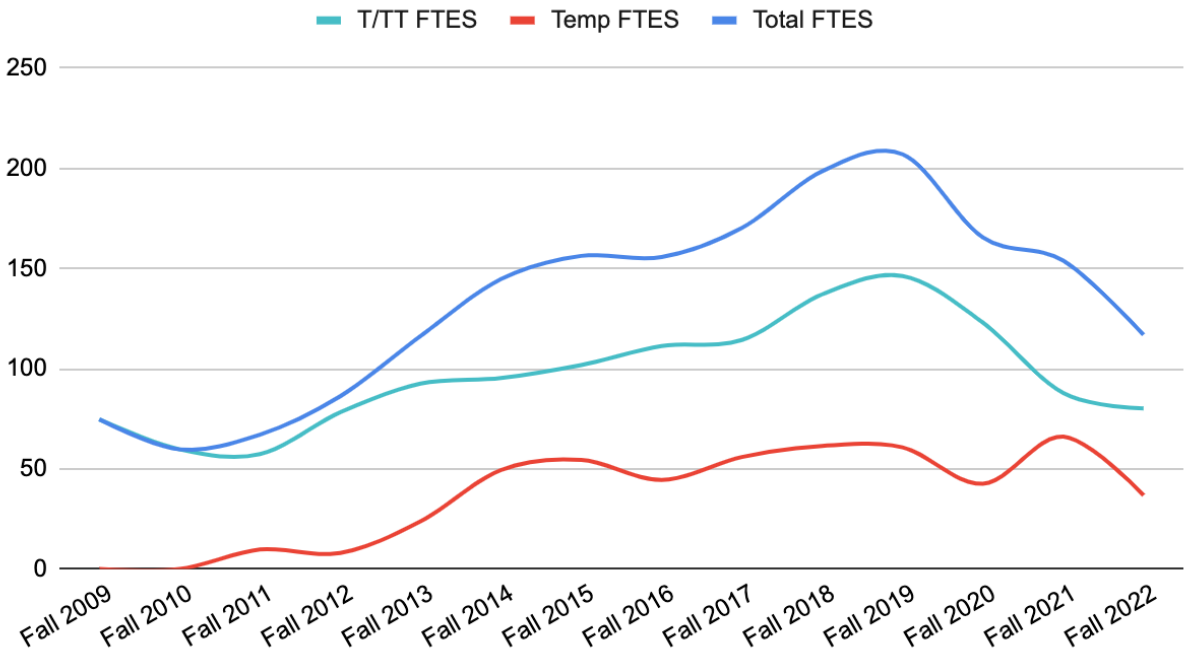

Figure 3.11. FTES for all CS classes excluding CS 101, 2009–2022

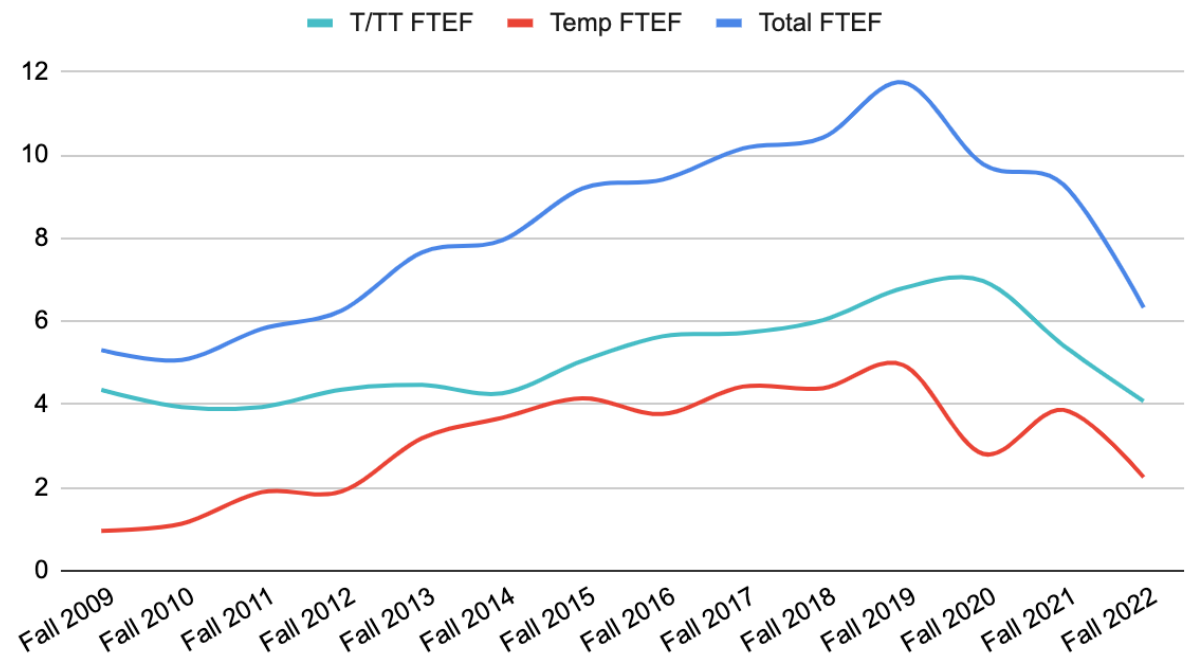

Figure 3.12. FTEF (averaged across Fall and Spring), 2009–2022

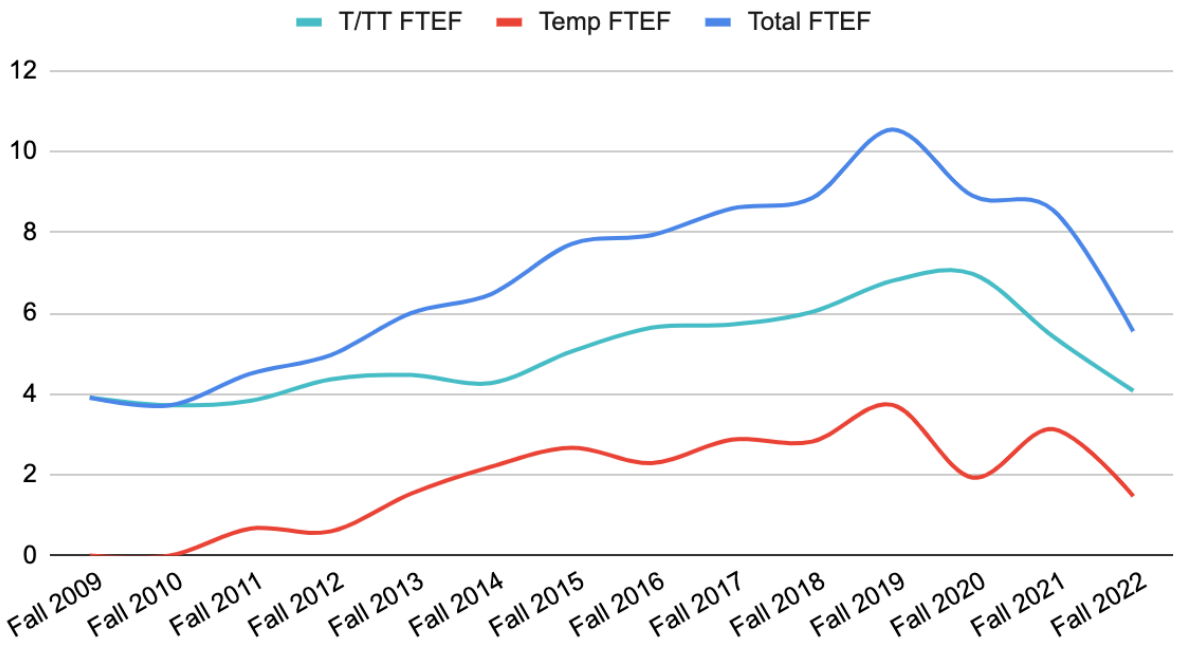

Figure 3.13. FTEF excluding CS 101 (averaged across Fall and Spring), 2009–2022

The student-to-faculty ratio (SFR) is FTES taught in a department divided by the FTEF used to conduct the instruction. Figure 3.14 reports this metric for the T/TT FTES and T/TT FTEF, showing this metric in terms of courses for the CS major delivered by permanent faculty in the department.

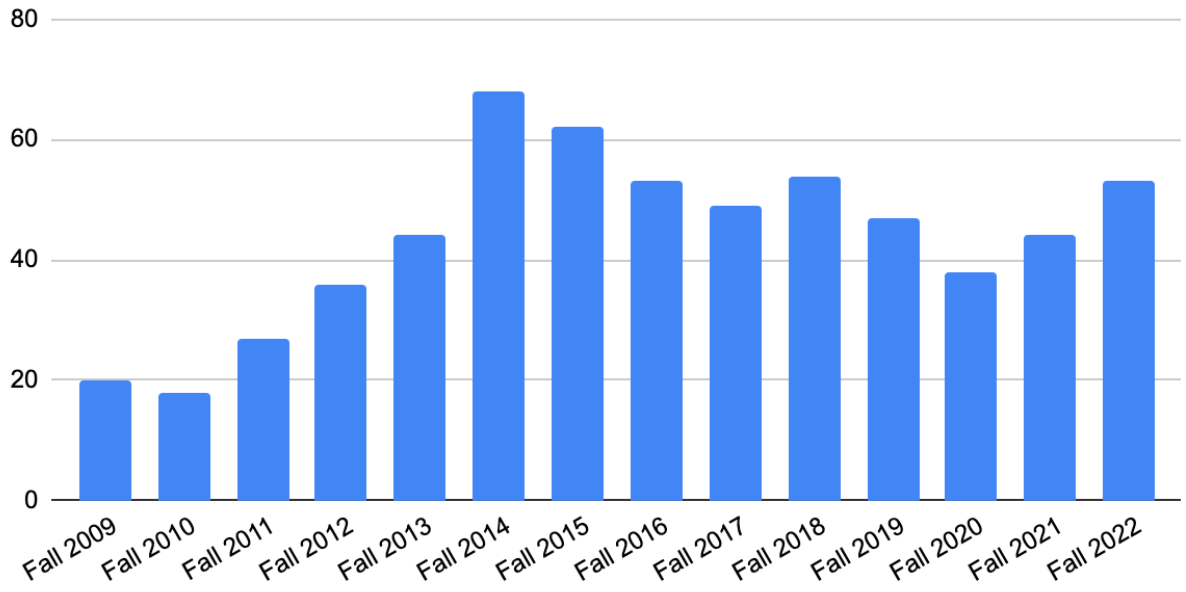

Figure 3.14. Student-to-faculty ratio (w.r.t. CS majors and T/TT faculty), 2009–2022

Another metric related to workload is a comparison of the contracted WTUs by T/TT faculty and the actual delivered WTUs by that faculty (see Figure 3.15). The difference reflects how much and how often permanent faculty have “volunteered” beyond their contract. As shown in Figure 3.16, this overage has been beyond 12 units, the total WTU for T/TT faculty. Thus, in aggregate, the department has been teaching so far beyond its contracted load that it has effectively manufactured a “phantom faculty member” to handle staff shortages. This phenomena is at least partially created by the fact that contract courses (directed studies, research capstones and internships) are not included in a faculty member’s labor plan (see Figure 3.17).

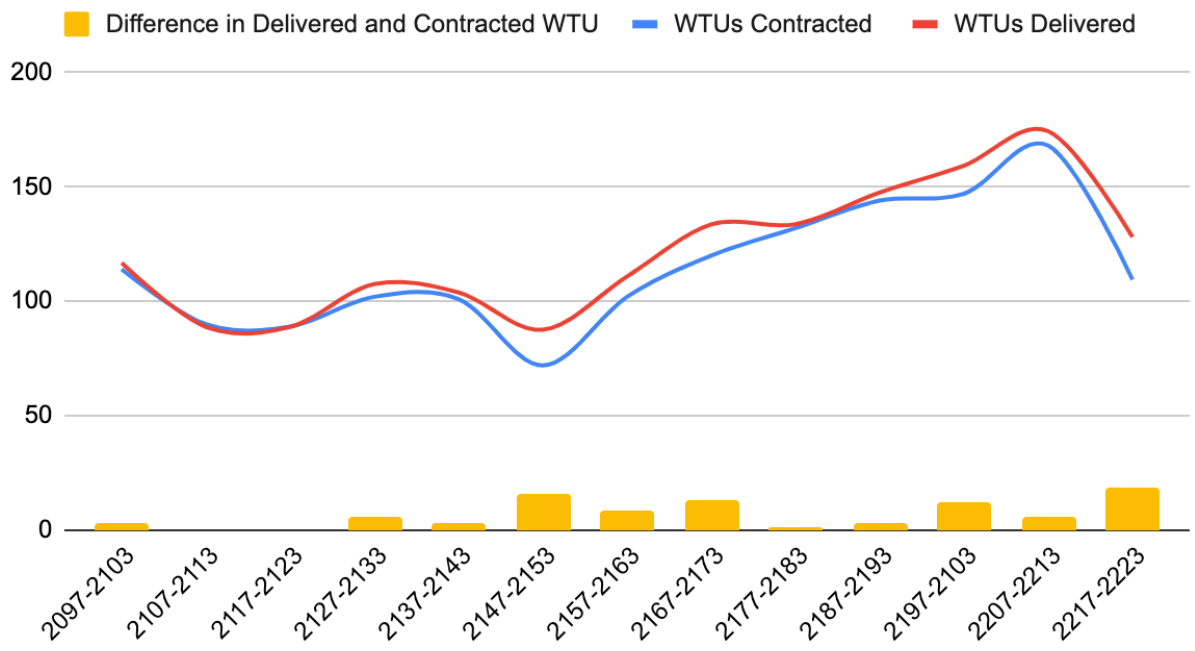

Figure 3.15. Contracted and delivered WTUs by T/TT Faculty, 2009–2022 <in faculty>

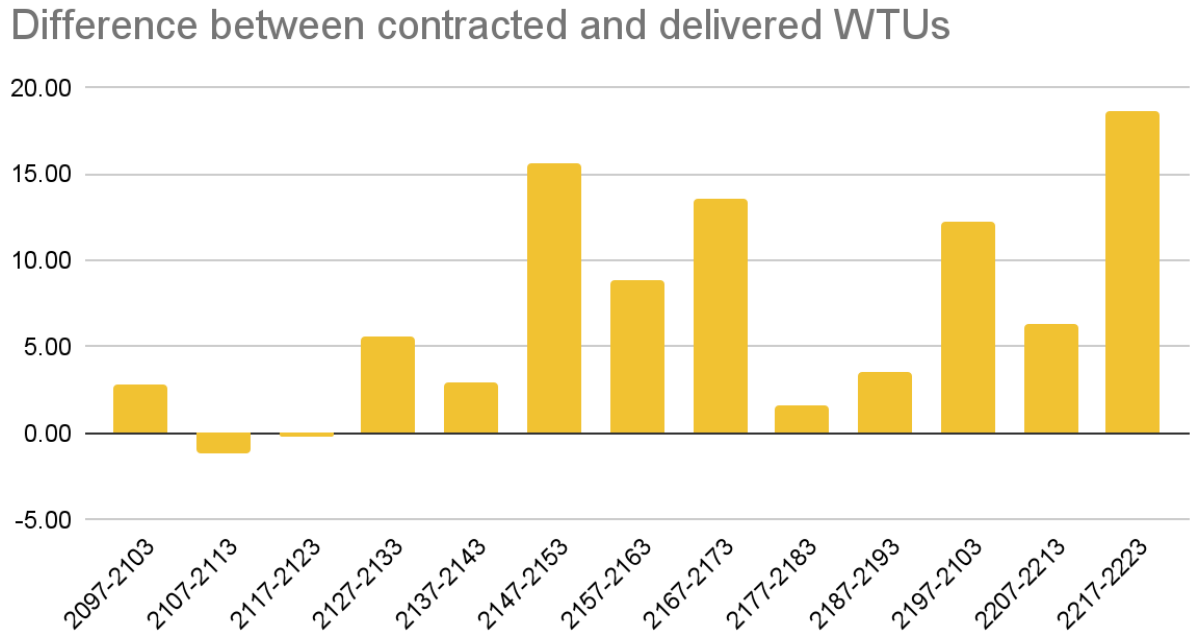

Figure 3.16. Difference between contracted and delivered WTUs (detail of Figure 3.15)

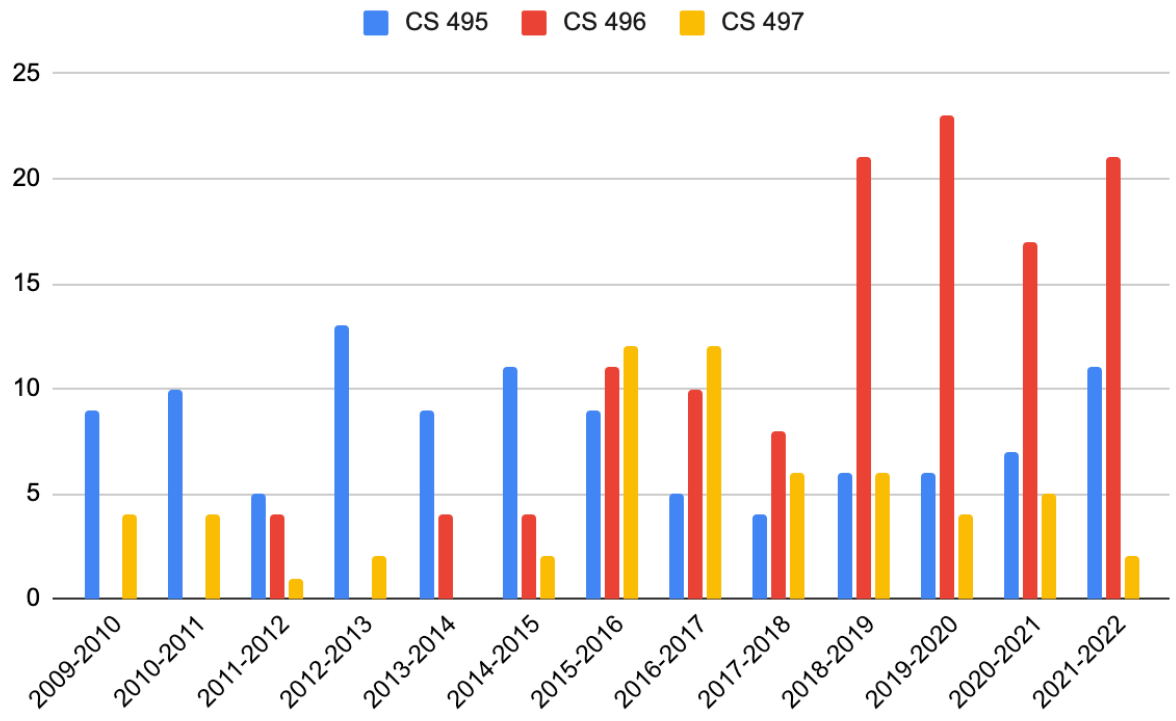

Figure 3.17. WTUs for directed studies, supervised research and internships, 2009–2022

3.3 Faculty Specializations

Gurman Gill (PhD, Electrical Engineering, McGill University, 2009)
Focus: computer vision, medical imaging
Research Brief:

> Dr. Gill's research revolves around application of computer vision and machine learning algorithms to multiple imaging domains such as medical, satellite, and microscopy. He regularly involves students in pursuing image analysis, image classification, and object detection tasks in inter-disciplinary fields such as remote sensing, and medical imaging. As a former game programmer, he instills software design, development, and debugging skills across the courses he teaches, which range from entry-level programming to game development to fundamentals of computer vision.

Mark Gondree (PhD, Computer Science, University of California, Davis 2009)
Focus: computer security, security education
Research Brief:

> Dr. Gondree's research is at the intersection of education, applied cryptography and system security. His research spans a wide range of application areas including operating systems security, network security, cloud security, embedded systems security, software security, multi-level security, and high-assurance systems. The focus of his

security education work is games for security education, including informal games (board games) and cybersecurity competitions (so-called cyber capture-the-flag games). He has released games in the cyber-security education field and, more generally, is an active contributor of OER to the computer science education field.

Ali Kooshesh (PhD, Computer Science, University of New Mexico, 1992)
Focus: design and analysis of algorithms
Research Brief:

Dr. Kooshesh's research and practice interests are in the areas of design and analysis of algorithms -- particularly geometric algorithms -- and large-scale, multi-paradigm distributed software systems. In the past decade, he has guided the development of several software products, some of which are being used on a regular basis in the department and shared with other departments across campus. His teaching interests range from the introductory programming courses to data structures, analysis of algorithms, operating systems, and the use of different programming platforms, such as full-stack web-frameworks and iOS-based mobile application development.

B. Ravikumar (PhD, Computer Science, University of Minnesota, 1987)
Focus: theory of computation, algorithm design and formal models
Research Brief:

Ravikumar has published over seventy-five papers in the areas of complexity, automata and formal languages and parallel computing with applications to program testing, image processing, machine learning among others. He has served on the program committees of numerous international conferences and is serving as an associate editor of the Journal of the Foundations of Computer Science. He has supervised several graduate and undergraduate students and has also served as an external examiner of Ph.D. students.

Suzanne Rivoire (PhD, Electrical Engineering, Stanford University, 2008)
Focus: computer architecture, high-performance computing, computer science education
Research Brief:

Dr. Rivoire's background is in computer architecture, both system-level and microarchitecture. Her main research focus has been in power- and energy-aware computing at scale, in both commercial data centers (in collaboration with researchers at HP Labs and Microsoft Research) and supercomputers (in collaboration with Oak Ridge and Lawrence Livermore National Labs). Her research encompasses both hardware and software characterization – in terms of modeling and benchmarking – and runtime systems to optimize performance under power and energy constraints. She has also contributed to the CS education literature on teaching parallel computing in the multicore era, tools for teaching quantum computing to CS majors, and Universal Design for Learning in the CS classroom.

Dave Shreiner (BS, University of Delaware, 1989)

Focus: computer graphics; real-time, parallel processing, and mobile computing; virtual reality & XR

Research Brief:

Dave is a 30-year veteran of industry having worked at companies like Silicon Graphics, Apple, ARM, and Unity Technologies, focusing on real-time image generation, virtual reality, and interactive applications. He's also the author of several books on graphics; contributed to the design of several industry-standard APIs including OpenGL, OpenCL, and others; and published several academic papers and awarded seven patents. Another area of interest includes computer graphics education, where he's co-authored a textbook on graphics, chaired the field's largest conference ACM/SIGGRAPH (2014) as well as their courses program several times, and has presented hundreds of short courses on graphics application development worldwide.

Henry Walker (PhD, Mathematics, M.I.T., 1973; MS, Computer Science, U. of Iowa, 1979)

Focus: Computer Science Education, Software Development, Algorithms

Scholarship Brief:

Dr. Walker has published 10 books and over 160 articles in his field. He is a prominent and enduring figure within the computer science education and liberal arts communities. Dr. Walker has been an MAA member since 1969, an ACM member since 1979, and a SIGCSE member since 1979. Dr. Walker's activities and career include:

- Current member of the MAA Committee on Program Review (2nd term on committee)
- Active member and co-founder of the new SIGCSE Committee on Computer Science in Liberal Arts Colleges
- External Reviewer of programs in computer science and/or mathematics: most recently, his 47th review (Pepperdine University, January 2023) and 48th review (SUNY---Purchase, scheduled for April 2023)
- Reviewer or Associate Program Chair for SIGCSE TS, ITiCSE, and many CCSC conferences for many years (over 20 years for several conferences)
- An ACM Distinguished Member and named by Distinguished Educator by ACM
- Served 6 years as Treasurer of SIGCSE, 6 years as Chair of SIGCSE, and was recognized in 2013 with an Award for Lifetime Service to SIGCSE
- Associate Editor and Columnist for ACM Inroads magazine
- Ongoing involvement in the Advanced Placement program, in both math and CS

3.4 Professional Development

There are both SSU-wide and CSU-wide professional development opportunities available to faculty, and CS faculty have historically made use of these opportunities. Often, CS faculty will benefit from professional development programs and, later, contribute to them as leaders, facilitators, authors, etc. The following are examples of resources from which CS faculty have benefited and to which they have contributed.

CSU Ensuring Access through Collaboration and Technology (EnACT) program was a US Department of Education-funded program to facilitate alignment of instruction with Universal Design for Learning (UDL) strategies. It was active 2009–2016. The EnACT Partnership, Technology & Dissemination (EnACT~PTD) Faculty Learning Community was a focal aspect of the program. At SSU, a CS faculty member was a participant in the EnACT~PTD FLC and then, later, served as facilitator for SSU's Universal Design for Learning FLC.

The CSU Course Redesign with Technology (CRT) program was developed in response to course bottlenecks that limited student ability to progress toward graduation, and was active 2013–2017. Multiple CS faculty participated in the CSU CRT program, contributing ePortfolios showcasing their year-long redesigns.

CSU Quality Online Learning & Teaching (QOLT) is a CSU-wide training program, related to the CSU Course Redesign with Technology Initiative. It supports faculty in improving the quality of online instruction using the Quality Learning and Teaching (QLT) evaluation instrument and the Quality Matters (QM) peer review process. CS faculty have participated in these programs (occasionally offered free or low-cost to SSU). CS faculty have also contributed to these programs, i.e., having teaching materials accepted into the CSU QLT's Quality Assurance Resource Repository (QuARRY) and performing QM peer reviews.

CSU's Affordable Learning Solutions is an initiative by the CSU Chancellor's office to promote the use of low- or no-cost course materials to reduce the financial burden on students. It has provided micro-grants to faculty to incentivize redesign and adoption of low-cost texts and OER, and CS faculty have participated. Some CS faculty have authored and contributed OER to MERLOT (a CSU digital repository of OER), and others have redesigned courses to be zero-cost.

CSU STEM-NET promotes research, community building and innovative educational ideas across the CSU university system. It provides support to faculty for new grant development, networking, and professional development opportunities.

The National Center for Women & Information Technology (NCWIT) Academic Alliance includes SSU as a partner. This affiliation brings with it recurring professional development opportunities. For example, NCWIT "Meeting of the Minds" is a webinar series geared toward postsecondary computer science faculty and those in student-facing roles.

The SSU Center for Teaching & Educational Technology (CTET)—formerly known as the SSU Faculty Center—hosts a variety of workshops and programming. CS faculty have participated in and contributed to faculty learning communities, professional development cohort programs, instructional workshops and other CTET programming, including:

- Participating in the “Berkeley STEM Faculty Learning Program” (STEM-FLP), based on UC Berkeley’s “Transforming STEM Teaching” faculty learning program (2016);
- Participating in the “Effective Teaching Practices” Faculty Learning Community cohort (2017), a pilot program using training materials from the Association of College and University Educators (ACUE);
- Participating in various course redesign programs responsive to the exigencies due to COVID (2019–2021), such as ACUE micro-credential programs, QM / QLT credential programs for online teaching, and SSU’s “Canvas Design Summer Institute”;
- Participating in the “Faculty Learning Community for Student Success” (2022)

CS faculty have also made use of a variety of external professional development opportunities, including the Professors Open Source Software Experience, POGIL facilitation and authorship programs, AP CS fellows programs, etc. Funding from the SST Dean, distributed via SST Professional Development grants awarded twice a year, can allow some faculty to take advantage of these external opportunities.

In general, funding and time are required to strengthen and grow professional relationships with strategic partners. SSU is part of the NCWIT Academic Alliance, but no faculty have had the opportunity to attend the annual NCWIT Summit. SSU is a member of the Computing Alliance of Hispanic-Serving Institutions (CAHSI), but no faculty have had the opportunity to attend any CAHSI all-hands meetings or affiliated conferences like Great Minds in STEM. No SSU department chair has attended the annual meeting of CS department chairs at CRA’s Conference at Snowbird.

3.5 Scholarship and Creative Activities

Details of faculty teaching, scholarly activities, service, and other efforts are provided in the CS faculty curriculum vitae in Appendix C. The following is a brief overview:

- Publications and and presentations: too many to list; refer to curriculum vitae
- Service to discipline: journal and conference referees; conference chairs; program and organizers; committee members; panelists and guest speakers; external PhD committee member
- Grants: multiple SSU Research, Scholarship, and Creative Activity Program (RSCAP) mini-grants; multiple Koret Scholar awards; multiple small travel grants for students; PI or co-PI on small grants from NSF, ORNL, etc; numerous grant proposals submitted.
- Faculty Awards: Excellence in Scholarship honoree; Excellence in Teaching Award nominees and honoree
- Student Awards: Awarded outstanding student club or organization (jointly awarded to CS Club & WiCS); Awarded outstanding student-organized program (NomaHacks, CS Club); SSU Science Symposium Bright Idea award winner; CSU Student Research Competition presenters and award winner
- Department service: Tenure Track Search Committees (2013-14, 2014-15, 2015, 2016, 2016-17, 2018-19, 2019-20, 2021-22); Lecturer Pool Refresh; Drupal Migration; Student Club Advisors (CS Club, Women in CS Club, among others).
- School service: members on multiple School RTP Committees; SST Professional Development Committee members; SST Elections Committee member; SHIP Coordinator; participant in Sonoma Mountain Connection grant (PIs Wade, Karp, Zippay); participant in TIPS grant program (PIs Lahme, Ford, Ortega).
- Campus service: Members and Chairs of numerous Faculty Senate standing committees and subcommittees (AAS, AFS, APARC, ATISS, EPC, GE, PDS, Scholarship, URTP, University Standards); departmental representatives and Executive Board Members of CFA; Facilitator of SSU's Universal Design for Learning faculty cohort; 2017 WASC self-study co-author.

4. Program Resources

4.1 Student Support

4.1.1 Advising

Prior to Fall 2012, the CS department chair performed all academic advising for computer science majors, minors advising, transfer advising, meeting with prospective majors/minors, and career advising. This became an unsustainable practice in light of program growth. Between 2013–2016, other faculty began to take on advising. Now, all tenured and tenure-track faculty in the computer science department participate in all aspects of advising majors and minors. Each student has an assigned academic advisor in the department and, if not, is advised by the chair. Given CS is a vertical major, academic advising necessarily includes both lower-division and GE advising. The student-to-advisor ratio over time (excluding minor advising) is shown in Figure 4.1, with the new advisor assignment strategy fully implemented by Fall 2017.

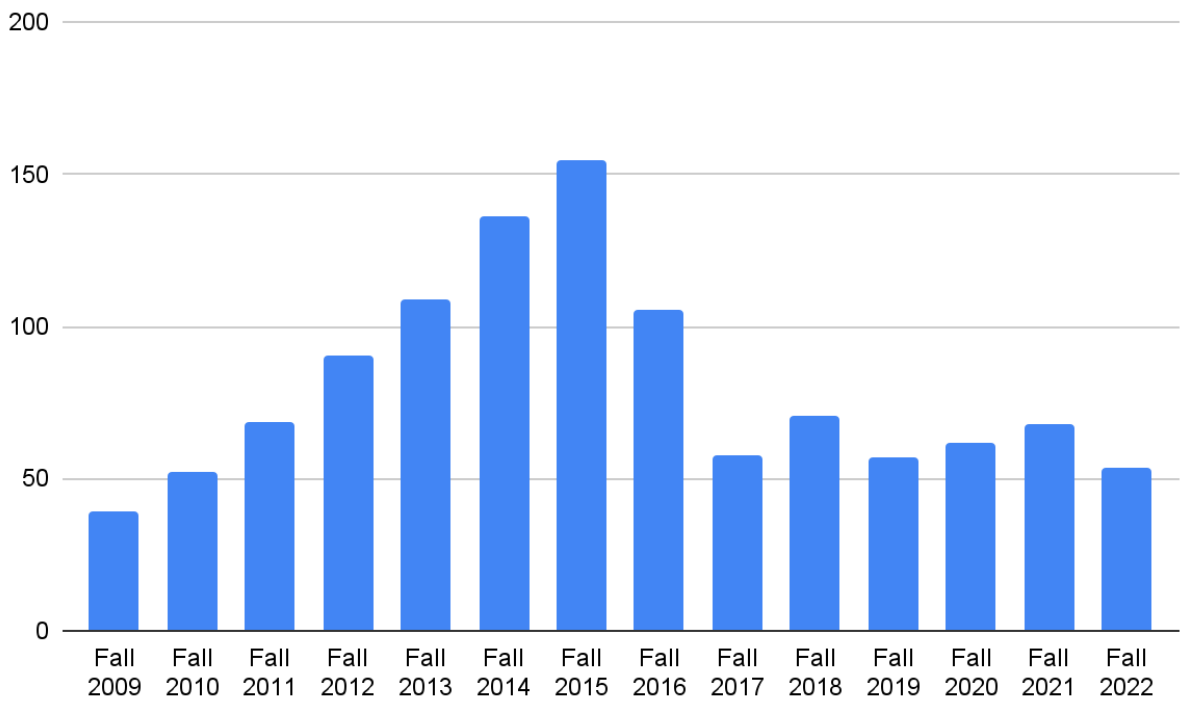

Figure 4.1. Student-to-Advisor Ratio since Fall 2009

In recent years, CS students have also received the support from a professional advisor specializing in a collection of STEM majors for early-career support. CS students may also have other advisors from different campus units (a MESA advisor, an EOP advisor, an athletic

advisor, a military and veterans resource center advisor, etc). This has required new tools (LoboConnect) to coordinate these networks and, sadly, has led to confusion among students.

Our advising strategy—balancing academic advising across full-time CS faculty and supplementing with staff advisors in other advising units—is employed by all comparable computer science programs across the CSU that we surveyed. Figure 4.1 shows the student-to-faculty ratio among our peer CSU institutions (again, excluding minors). Our SSU computer science program has the most unfavorable student-to-faculty ratio among our peers. This has repercussions beyond academic advising.

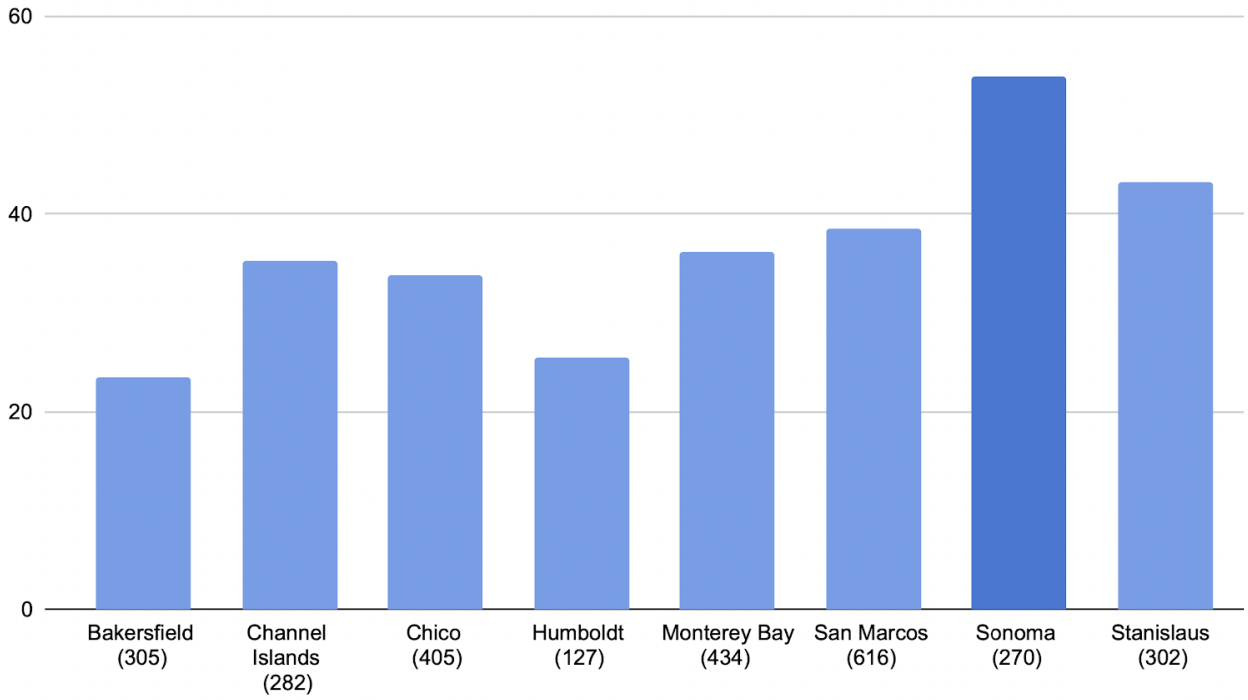

Figure 4.2. Comparison of student-to-faculty ratio with peer CSU campuses, Fall 2022

While the CSU/CC system has robust transfer pathways and course articulations, no two academic programs are ever in perfect alignment, by any standard. This is especially true in computer science programs, given the field itself is relatively young and very dynamic. In CS, a large number of transfer students are admitted to the University and register at relatively unpredictable times. Transfer advising becomes a complex, time-sensitive task. Placement of transfer students is difficult when classes are full and when some lower-division coursework is missing.

4.1.2 Campus Support Services

The SSU student community benefits from a variety of campus-wide support resources, including:

- Campus-wide IT services (see Section 4.3 for a related discussion)
- Student Affairs' Basic Needs Initiatives (Lobo's Pantry, etc)
- Counseling and Psychological Services (CAPS)
- Disability Services for Students (DSS)
- Campus Recreation Center's fitness and enrichment programs
- Student Career Center and Job Fairs
- Student Involvement support for student clubs and activities
- Learning and Academic Resource Center (LARC) tutoring services
- Mathematics Engineering Science Achievement (MESA) Program at SSU

The computer science students benefit, in particular, from many of these services. For example, several CS classes receive tutoring support through both the LARC and MESA tutoring programs. The CS colloquium receives support via Instructional Related Activities (IRA) funds. Student Involvement supports both the Women in Computer Science club and CS club.

The MESA program is part of a nationally recognized California academic support program to increase the number of historically underrepresented students in STEM.

Over the period of this review, the MESA program at SSU has seen tremendous growth. It serves CS students, however, in disproportionately-low numbers: the percentage of CS students involved in MESA is less than half what might be expected if it served all its target STEM programs, in proportion (see Table 4.3). At other CSU campuses, the MESA program tends to serve engineering and CS students in much greater proportion than other STEM majors (some MESA chapters serve engineering and CS, exclusively). Indeed, the CS department was one of the founding members involved in bringing the Mathematics, Engineering, Science, Achievement (MESA) program to the School of Science and Technology in 2009, when it was called the MESA Engineering Program (MEP) and CS Department Chair Dr. Statuffer served as its director. We believe there are opportunities for MESA to do more to partner with CS in programming to serve its students.

Table 4.3 MESA membership data, from the SST Spring 2023 meeting (Appendix F)

Major	MESA members	SST majors
Biology	49.2%	42.0%
Computer Science	**10.3%**	**25.8%**
Engineering	8.5%	10.7%
Mathematics & Statistics	17.5%	9.8%
Chemistry	7.9%	5.2%
Geology	1.6%	3.0%
Physics & Astronomy	4.0%	3.5%

There has been no systematic investigation by the CS department to evaluate which campus services our majors view as valuable or less valuable. Anecdotally, the programming and services of the Career Center do not appear to meet the expectations of our majors. The response by the department has been to develop resources to help address their needs (including the 1-unit Computing Professions course, CS 391). Likewise, our student clubs arrange their own industry trips, find CS alumni to run mock interviews and resume writing workshops, etc. Periodically, faculty advisors have informed each club that the career center may help run the type of professional development programming they organize as club events; however, no enduring relationship has been established by either party. Student feedback of campus services should be considered as part of any on-going evaluation.

4.1.3 Student research and/or community engagement

CS faculty participate regularly in research opportunities at the institutional-level and school-level, including (but not limited to) mentorship and research under the following programs:

- Research, Scholarship, and Creative Activity Program (RSCAP)
- Koret Scholars research projects
- Louis Stokes Alliance for Minority Participation (LSAMP) research projects
- Genentech Fellows Summer Undergraduate Research through MESA
- Summer High School Internship Program (SHIP) program through SST
- Center for Environmental Inquiry programs (e.g., WATERS)
- CalBridge and McNair Scholars programs

One barrier to greater participation in these programs is faculty availability. In particular, faculty support of student research through contract courses (CS495, CS496, CS497) is treated by the school as either service or uncompensated labor, because its WTU credit is never made part of faculty labor plans. We are unsure how contract courses are handled in other schools at SSU or at other CSU campuses.

In spite of the workload barrier, faculty support student research to the benefit of both students and SSU. This work is featured prominently in the final weeks of each semester's CS Colloquium, in the Science Symposium during SSU's Week of Research and Creativity, in the CSU Student Research Competition, and at disciplinary-specific conferences (see the Faculty CVs in Appendix C for many of the results of the research performed with students).

4.2 Library and Information Resources

The information needs of our students and faculty fall into two broad categories:

1. Resources for computer science practitioners, focusing on programming languages and software development infrastructure.
2. Resources for scholarly research and service, centering on access to publisher repositories.

As the capabilities and applications of technology change, CS practitioners must frequently adapt to new languages and tools throughout their careers. The literature shows that software practitioners typically are expected to acquire these new skills as needed on the job, without formal or explicit instruction [Cai19, Chakraborty21, Shrestha20]. To do so, developers generally make use of freely available informal resources such as online documentation, tutorials, videos, and Q&A sites, through a process of information foraging [Pirolli95, Ko06]. As our students make the transition from novices to experienced programmers, they too begin using freely available resources to pick up the new tools they need for specific courses or projects. Our faculty, rather than library faculty, are their mentors and guides in this process.

For scholarly research and professional service, on the other hand, access to traditional publisher repositories is essential. Over the past five years, we have had consistent access to the most important non-free repositories for computer science: the Association for Computing Machinery (ACM) Digital Library; the Institute of Electrical and Electronics Engineers (IEEE) Xplore; and much of Springer's vast corpus of conference proceedings and monographs. Our access to those resources has generally been improving since the prior program review in 2007-08, with CSU systemwide deals with publishers ensuring access even during campus financial downturns.

In summary, we require only a few specific information resources from the University to support our department's curriculum and scholarship, and these needs have been met in recent years.

4.3 IT Resources

Campus-wide information technology support for instruction and instructional spaces is primarily divided between two entities: the IT Department and the Center for Teaching & Educational Technology.

4.3.1 Technology resources for classroom instruction

Standard classroom technology at Sonoma State University consists of whiteboards, projection screen, a document camera, a digital projector with HDMI support, and a computer workstation at a lectern (see Appendix G for details). Some classrooms have reconfigurable seating for students that is more adaptable for group work and active learning. There are currently no classrooms supporting distance education, hybrid/hyflex instruction, or recording capabilities. The SSU IT department had previously provided student staff support to record select classes and speakers and the CS department benefited from this service (ita Colloquium series was regularly recorded and featured on SSU's YouTube channel). In the past five years, this service is no longer provided, presumably as a cost-cutting move.

The technology serving campus instructional spaces is periodically reviewed by the Senate's Academic Technology and Instructional Spaces Subcommittee (ATISS). The CS department's instructional spaces are not recommended for upgrades and not subject to ATISS's campus-wide reviews, and they are not considered general-purpose classrooms. As a result, the instructional technology in general-use classrooms tends to be of higher quality than CS instructional spaces.

4.3.2 Technology resources for student work

While there are computer workstations for student-use at several campus locations—including the library, the LARC tutoring space, and the 24-hour computer lab—these do not support the software required for computer science coursework, such as software development tools. Several student support programs exist to provide technology resources to students in need, including the library Laptop Loan program, the CSUCCESS (California State University Connectivity Contributing to Equity and Student Success) program, and the LSAMP (Louis Stokes Alliances for Minority Participation) program. These are each wonderful programs but fall short in their equity goal to equip their target students with adequate resources to be able to succeed in their college career. In particular, these programs generally provide students with an iPad, Chromebook, or some locked-down device without the ability to install any software development tools required in their CS classes. CS students eligible for these programs and able to receive its technology resources have often returned them, due to their inadequacy in their CS coursework. Given the lack of campus technology resources supporting the most modest of CS coursework, this demand is partially fulfilled by CS department facilities playing double-duty as much needed open lab space capable of supporting CS coursework (see Section 4.4). Further, students have reported trouble connected to CS server resources from the

on-campus residence halls. The network design and firewall rules managed by IT are opaque to the department and change without coordination with campus stakeholders.

4.3.3 Technology resources for course development and delivery

The SSU Center for Teaching & Educational Technology (CTET)—formerly, the SSU Faculty Center—supports faculty excellence in teaching and learning through personalized consultations and targeted workshops on how to best use technology to improve teaching. They support faculty to develop online courses, use campus resources for instructional technology like Canvas, the Google Suite products, Camtasia and other software available through campus-wide licenses. CS faculty have both benefited from and contributed to CTET programs (see Section 3.3).

4.3.4 Technology resources for advancement of instruction / research / scholarship

Computing resources for research and education on campus are relatively limited. These include AWS educational credits, supercomputing allowances via our campus participation in the NSF XSEDE/ACCESS programs, and other small grant-supported programs.

Our ability to participate in cyber infrastructure and advanced computing programs are complicated by CSU procurement rules and, in particular, SSU's local interpretation of those rules. The contract process is required even for free services and open-source software, as each still requires a zero-cost contract to be reviewed through procurement. Further, most off-site computing resources qualify as either Software-as-a-Service or Infrastructure-as-a-Service, and thus fall under rules related to the procurement of cloud services. The SSU procurement office appears to have little capacity for certifying/clearing software and platforms for classroom use under these rules. This is made worse by the fact that every contract renewal is treated as a new contract. SSU also appears unable (or unwilling due to risk) in leveraging prior procurement research undertaken by other CSUs to their advantage. One particular hurdle impacting most services is the requirement that they use SSU single-sign-on (SSO) to limit the Type-1 data managed by that service. This is made complicated by the fact that no transparent resources exist to set-up and configure systems to utilize SSU SSO through a safe, transparent, standards-based API. This is true even for services run locally at SSU by professors. Configuring SSU SSO services to work with any new service is perceived as a significant staff investment, requiring a reassignment of IT personnel with business justification.

In summary, heavyweight procurement processes and lack of a clear business process for supporting services via campus SSO have impacted CS faculty adoption and development of technologies for classroom instruction, for homework collection and unit testing, for interactive textbooks, and for collaboration across the CSU on research programs.

4.4 Facilities and Instructional Spaces

Since 2006, the Computer Science Department has operated three instructional laboratories and a server facility housed in the basement of Darwin Hall. Since 2006, the number of new CS majors and the need for instructional spaces for classes serving them caused the department to grow beyond the capacity of these three labs. In Fall 2016, the department expanded its set of instructional spaces to include a fourth lab, outside the Darwin basement, in Stevenson Hall.

Table 4.4. Instructional Spaces for CS, Fall 2009 – Fall 2022

Building & Room	Instructional Space
Darwin 24	Computer Literacy and Multimedia Lab
Darwin 25	Software Design Lab
Darwin 28	Advanced Computing Lab
Stevenson 1034	Computing Instruction Lab

The primary facilities comprising the non-instructional spaces for CS have remained stable over the period of review. These include CS faculty offices, a storage closet, and a basement office (Darwin 26) that has served, at alternate times, as a faculty project space, student club office, and shared lecturer offices.

Table 4.5. Non-instructional Spaces for CS, Fall 2009 – Fall 2022

Building & Room	Role
Darwin 116	Department office and offices for faculty / lectures (116A, 116B, 116C, 116D, 116E, 116F, 116J, 116H, 116I)
Darwin 27	CS Tech (Sysadmin) Office
Darwin 26A	CS Server Room
Darwin 26	CS Office space
Darwin 39	Storage Room

4.4.1 Spaces supporting CS Instruction

Each instructional space plays more than one role in the computer science department, acting as (i) classroom for CS courses; (ii) tutoring space for CS tutors; (iii) club space for WiCS and CS club; (iv) open lab space for CS students when not in use by classes; (v) workspace for student projects and capstones; (vi) an occasional general-purpose lab for SST and SSU (used by Expanding Your Horizons outreach activities, AC trainings, etc).

No CS instructional lab is large enough to support both a lecture/active learning part and a lab part. The rooms are arranged with workstations in rows, optimized for room capacity and facilitating individual work. These instructional spaces do not facilitate peer programming / active learning under the current room configuration and rooms are not re-configurable. No CS instructional spaces have infrastructure usable in classes by students, such as air-gapped LANs for student-managed servers and/or networking equipment (for students to learn to install and manage infrastructure).

Each lab includes AV equipment for slide projection onto one classroom wall. This AV has not been refreshed by SSU since the Darwin renovation in 2006. This AV is no longer serviced by the campus IT as the equipment has been deemed “unsupported” due to obsolescence. The CS department has periodically managed to swap some of its AV equipment with “general assignment classroom salvage” (obsolete-and-unsupported equipment being excessed during equipment refreshes that excluded CS spaces.) It is unclear why CS spaces—essential to instruction in our program—have never been included in any campus-wide equipment refresh.

Darwin 26A

The server facility for the department is located between Darwin 25 and 28. It houses a variety of equipment, but the most significant server resources are:

- Server blue.cs.sonoma.edu, last refreshed June 2018.
  HPE ProLiant DL380 Gen10/ProLiant DL380 Gen10, BIOS U30. It currently has Intel(R) Xeon(R) Gold 5115 CPU @ 2.40GHz 2 CPU, with 10 cores each, with 128Gb of RAM, and 11 TB of RAID 5 disk for user files.

- Server kodiak.cs.sonoma.edu, last refreshed Oct 2015.
  The backup server to blue.cs.sonoma.edu, an HPE ProLiant DL380 Gen9. It has a current backup of the user file space, and most of the software that is installed on blue.cs.sonoma.edu.

- Server kirby.cs.sonoma.edu, last refreshed March 2013.
  An older version of kodiak.cs.sonoma.edu, an HPE ProLiant DL380 Gen8. It has fewer cores and less userfile space. It has a Raid Disk for data storage of large data sets.

Most courses in the CS major utilize blue.cs.sonoma.edu for instruction and student projects. Notable courses making heavy use of the this server include Intro to Unix (CS 210), Data Structures (CS 315), Database Management Systems Design (CS 355), Intro to Computer Security and Malware (CS 340), Operating Systems (CS 450), among others. It is the de facto environment for testing and evaluating student coursework in the department.

Darwin 24

The Computer Literacy and Multimedia Lab is mainly used for hands-on instruction for CS 101. The lab holds 24 workstations. These machines were most recently refreshed in Dec 2017.

Each workstation is a 21-inch iMac computer (iMac18,2 2017), which is an Intel-based iMac (3 GHz Quad-Core Intel Core i5) with 16GB of memory; 1TB hard drive; Radeon Pro 555 2GB graphics card; and a Retina 30-bit color display.

Darwin 25

The Software Design Lab is used mainly for Computer Science instruction. The lab holds 24 workstations. These machines were most recently refreshed in Nov 2022.

Each workstation is a 24-inch iMac M1 computer (iMac21,2 A2438 - 2021), which is an Apple M1 chip with 8 cores (4 performance and 4 efficiency); 7-core GPU supporting Metal GPUFamily Apple 7; and a 4.5k Retina display.

Darwin 28

The Advanced Computing Lab is used mainly for Computer Science instruction. The lab holds 24 workstations. These machines were most recently refreshed in Jan 2019.

Each workstation is a 21-inch iMac computer (iMac18,2 2017), which is an Intel-based iMac (3 GHz Quad-Core Intel Core i5) with 16GB of memory; 1TB hard drive; Radeon Pro 555 2 GB graphics; and a 4k Retina Display.

Stevenson 1034

This lab is used mainly for Computer Science instruction. The lab holds 24 workstations. These machines were most recently refreshed in Jan 2016.

Each workstation is a 21-inch iMac (iMac16,2 - Late 2015), which is an Intel-based iMac (2.8 GHz Quad-Core Intel Core i5) with 16GB of memory; 1TB hard drive; Intel Iris Pro Graphics 6200 supporting MacOS GPUFamily1 v4 Full High Definition 1080p display.

The Stevenson instructional space was unavailable in Spring 2020 due to the pandemic and Stevenson building remodel. Whereas other departments impacted by the remodel were given storage or a location to which to retreat, CS was given the task of finding its own facilities to store its Stevenson lab equipment for the duration of the remodel. It was given no adequate replacement space for instruction during the remodel. It was not allowed to retain the furniture that supported the Stevenson workstations (consisting of “pop-up” monitor desks with mounting arms to support each workstation display). CS was given no support or lab refresh funds to assist moving into its post-remodel Stevenson space (Stevenson 1210). CS was not involved in the layout of the new Stevenson lab space and the furniture in that lab cannot support the existing workstations (purchased in 2016, in-storage since 2020). As a result, this space is currently unprepared to support CS instruction.

4.4.2 Faculty offices & meeting spaces

The CS Department main office is housed on the first floor of Darwin in a suite of nine offices used for tenure-track and temporary faculty. It includes a mailroom and front desk for an AC. It is generally adequate for faculty and staff operations.

There is no dedicated CS space for department meetings. Meetings are organized by reserving a conference room in one of several campus locations. There are no dedicated spaces for department social gatherings outside the CS lab instructional spaces.

4.4.3 Research & scholarship facilities

There continues to be no dedicated space for CS faculty research, student capstone projects, and student research. This has impacted the ability of faculty to work collaboratively with students under the auspices of a summer REU or grant-supported research. Instead, students work independently privately at home or in public lounges on campus, meeting with their mentor in the faculty's office. Lack of spaces for scholarship and collaborative projects may make SSU less attractive to new tenure-track faculty eager to start a research career and work with students outside the classroom.

4.4.4 Student study & social spaces

There are no student study spaces in the department outside the CS lab instructional spaces. Students are encouraged to request key access from the department and use these labs during normal building operating hours, when otherwise not in use. Historically, access to these spaces by students after-hours and during campus holidays has been a problem.

In our last program review, our external reviewer recommended that "*[i[nformal interactions between faculty and students need to be facilitated through closer proximity between faculty offices and student laboratories*" and "*[t]he lack of a student work space that is near faculty offices and limited availability of laboratory resources outside of scheduled class times (open lab) impact student retention and program cohesiveness*" (CS Program Review Report, Sept 2008, Appendix A). No strategic reorganization of space within Darwin Hall resulted. In fact, the expansion of the department into Stevenson lab space—while crucial for scheduling CS classes to meet student demand—has worsened this fragmentation.

4.5 Staff Support

Over the period covered by this review, the CS Department staff has consisted of one (shared) administrative coordinator and one (full-time) system administrator. The Office of the Dean for School and Science & Technology serves as the direct supervisor of all administrative coordinators in the school. Working with the Dean's office, the department has always benefited from part-time AC support over the period of this review, summarized below.

Table 4.6. Summary of Administrative Coordinator Support since Fall 2009

Approximate Period	Total Staff	AC	Shared with	Avg program FTEs during Period
Spring 2007 – Winter 2010	0.5	Gina Voight	Geology	~131
Spring 2010 – Spring 2011	0.5	Jen Aaseth	Geology	~118
Spring 2011 – Summer 2012	0.5	Cory Oates	Geology	~126
Summer 2012 – Summer 2016	0.5	Liz Kettman	Geology	~189
Summer 2016 – Summer 2021	0.5	Dena Peacock	Geology	~217
Fall 2021 – Present	0.5	Kate Lapp	EE	~155

The visible trend in Table 4.6 is non-increasing staff support in the face of program FTE growth. The less visible trend is an overall reduction in the number of operating hours of the CS office. Since 2012, a reduction in staff hours due to personal schedules, personal preference and/or personal emergency have consistently resulted in either the early closure or unexpected closure of the CS department, with a sign referring requests within business hours to other offices (usually, the adjacent office of the Dean). Currently, the CS office is open/staffed three afternoons a week.

Our system administrator, Mr. Roger Mamer, has many duties and responsibilities within the department, including supporting all student workstations in CS instructional spaces, CS department's servers, and CS faculty workstations. While primarily charged with supporting our department, Mr. Mamer is often called upon to help other SST departments located in Darwin Hall (Biology, Chemistry, Geology, Mathematics, and Physics & Astronomy) and assist other lab technicians in the School of Science & Technology. The demand for academic computing and technology support within the school, inside and outside of the Computer Science Department, is high and requires substantial support. Mr. Mamer also supports the occasional use of CS computing labs by the wider campus community when they run into campus lab availability limits, such as during summer orientation.

5. Student Success

5.1 Student Enrollment and Demographics

In Fall 2022, the CS Department had 270 majors, 25 minors, and 2 post-baccalaureate (second Bachelor's) students. Among the School of Science and Technology (SST) departments, this is the fourth highest number of majors, just behind Nursing and Kinesiology, and more than 150 students ahead of the fifth largest department.

Figure 5.1 shows the historical enrollments of SST programs by department from Fall 2012 through Fall 2022. With the exception of Biology, which has recently removed its secondary admissions criteria (i.e. become un-impacted), every SST department has seen declines in enrollment in the past three years corresponding to the larger university-level trends. Computer Science mainly stands out for our growth between 2012 and our peak in 2018, corresponding with national disciplinary trends rather than university-level enrollment trends. The net effect is that we are the only program that is still markedly above its 2008 enrollment levels.

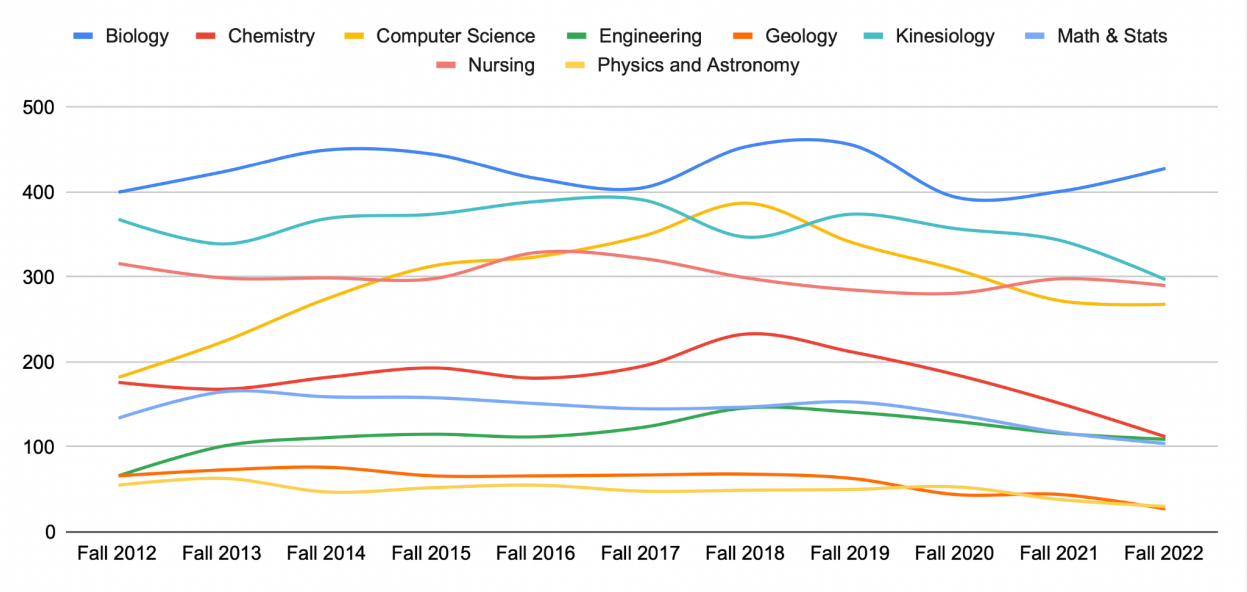

Figure 5.1. Number of enrolled majors in the nine SST departments, Fall 2012-Fall 2022. This graph, unlike other graphs in this section, only goes back to Fall 2012 because it comes from a different data source than the purely internal CS department statistics shown later in this section.

The CS department is also one of just a handful of programs whose enrollment declines seem to be stabilizing, although it is too soon to be certain. Our enrollment decline began more steeply than both SST's and the University's, which we would attribute partly to our "capacity crisis" creating instability in our course offerings, limiting our ability to adequately advise and even teach our students, and preventing outreach to high schools or feeder community colleges. However, in this academic year, our enrollment declines leveled off while the University's larger enrollment declines continued, as Figure 5.2 shows.

Figure 5.3 gives the historical breakdown of our students by program (majors, minors, and post-baccalaureates) since Fall 2009. The vast majority of the students we serve are majors, which was also the only population to see the volatile enrollment surge and decline of this period.

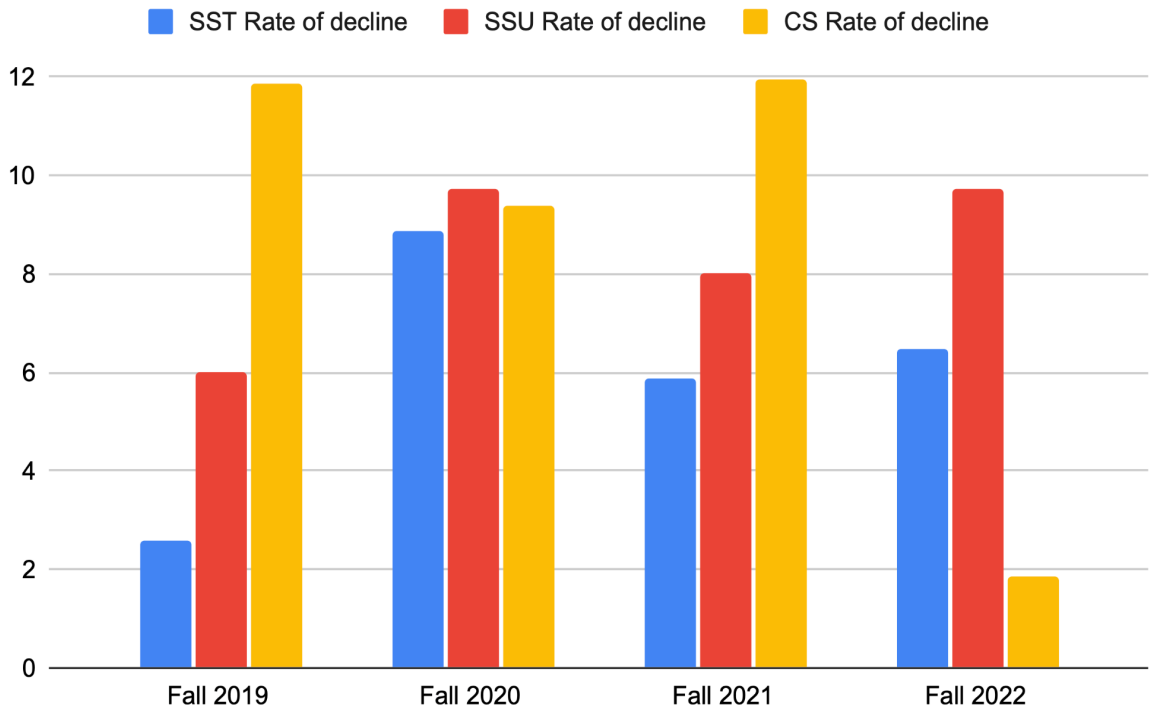

Figure 5.2. Rate of decline starting in Fall 2018, compared to previous year. Higher numbers represent worse declines.

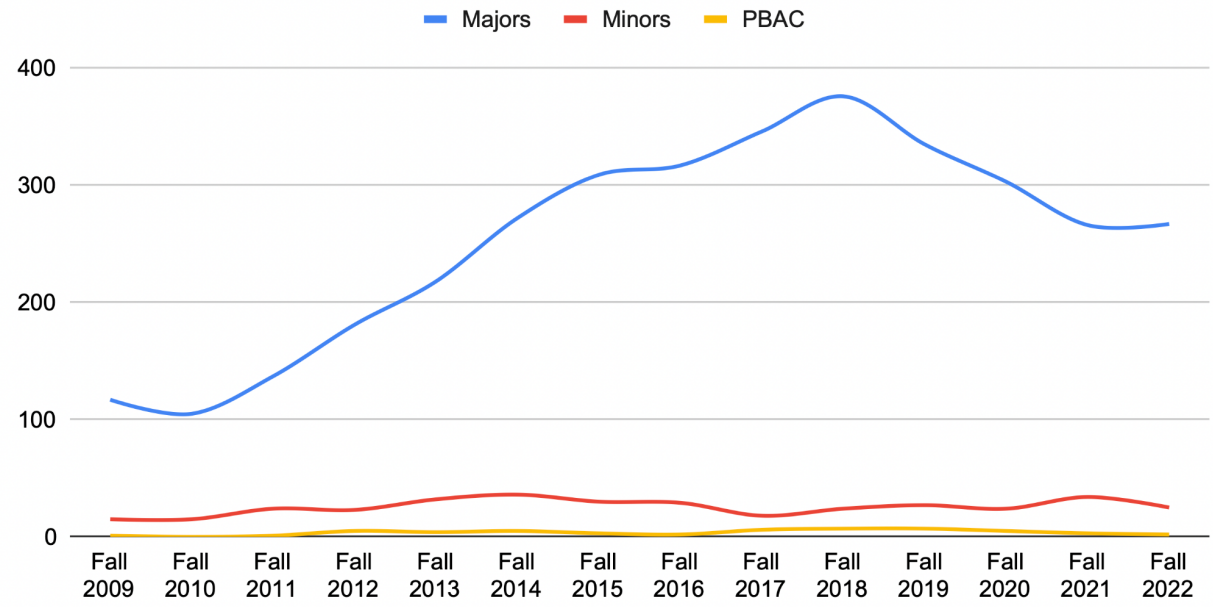

Figure 5.3. Number of SSU CS majors, minors, and post-baccalaureate students, Fall 2009-Fall 2022.

5.1.1 Student Demographics: Gender

One of the notable challenges that almost all Computer Science programs face is the low percentage of female students. The percentage of female students in our program was 15% in Fall 2022. Over the review period, it has fluctuated between 13 and 19 percent, as Figure 5.4 shows.

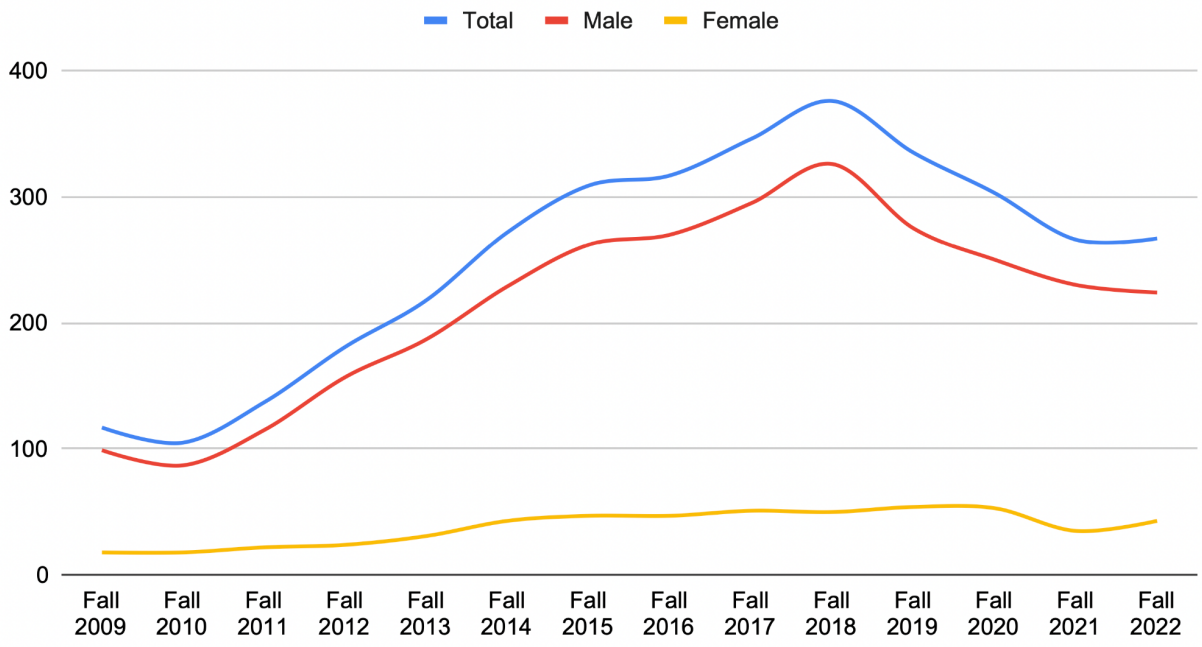

Figure 5.4. Total number of SSU CS majors with gender breakdown, Fall 2009–Fall 2022.

Nationwide in 2021, the percentage of female undergraduate CS majors at North American Ph.D.-granting universities stood at 21.9 percent [Zweben22, table B8]. However, it is worth noting that these research universities have considerably more international students than the CSU. International student CS majors are more likely to be female than Americans, so we would expect CSUs to have lower numbers. This is indeed the case; Figure 5.5 shows the percentage of female students in comparable (undergraduate-focused) CSU CS programs, all of which are below the research universities' 21.9% figure.

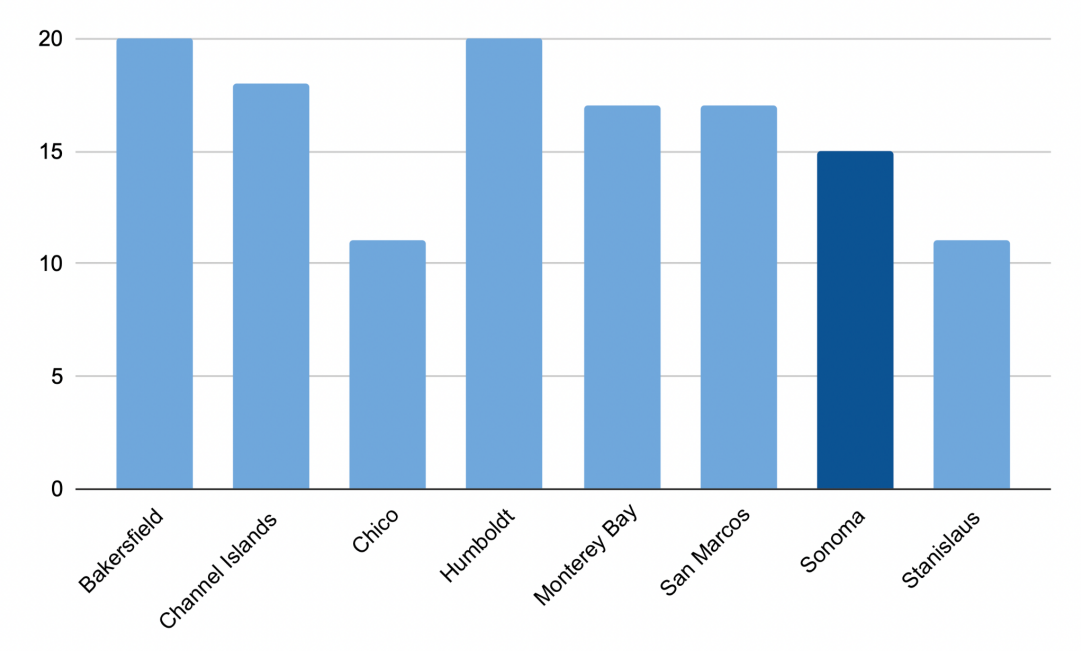

Figure 5.5. Percentages of female students in comparable CSU Computer Science programs, Fall 2021.

Providing support to our existing female students, as well as recruiting new ones, is a major area of concern for the department. Of our recruitment and outreach efforts, historically the most successful has been the faculty and student panel in CS 101, our GE/nonmajors offering whose demographics are closer to that of the overall SSU campus. For decades, a panel of female faculty members and representative CS majors have attended one of the CS 101 lectures each semester and participated in a panel about our program and their experiences. At one point, nearly half of our female majors came to us via CS 101.

5.1.2 Student Demographics: Ethnicity

Unlike the representation of female students, the percentage of underrepresented minority (URM) students in our program has nearly tripled between Fall 2009 (11.97%) and Fall 2022 (33.71%), as shown in Figure 5.6. This is due to a dramatic increase in the Hispanic population of both our major (Figure 5.7) and the University as a whole.

The ethnic demographics of our program more closely track SSU than the demographics of undergraduate CS programs at Ph.D.-granting institutions, as Table 5.8 shows. Most notably, while the percentage of our majors who are Asian has increased slightly over time, it is much closer to the relatively low numbers for the University than the much higher national numbers.

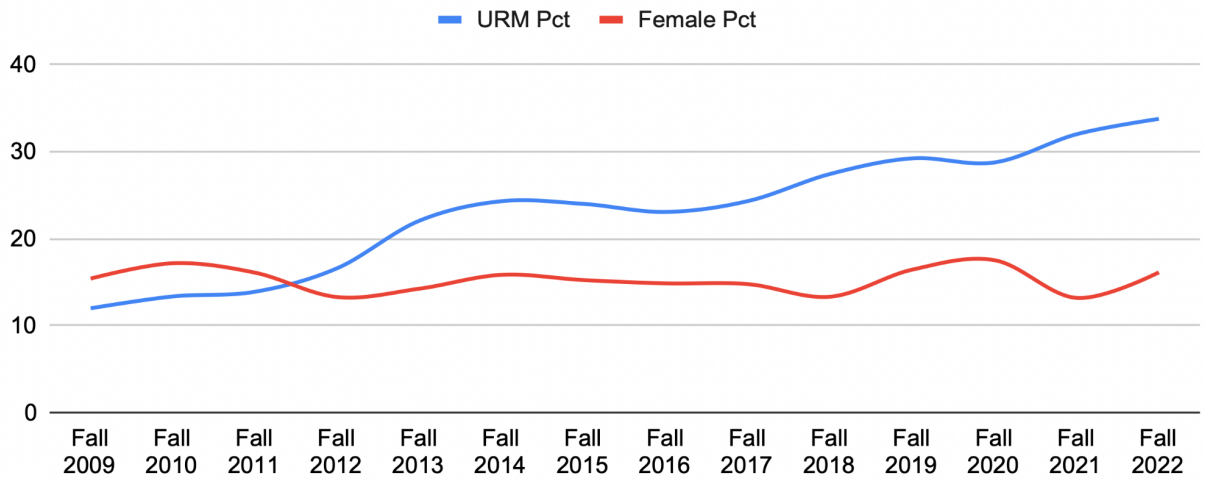

Figure 5.6 Representation of female and underrepresented minority students among SSU CS majors (as a % of total majors), Fall 2009-Fall 2022.

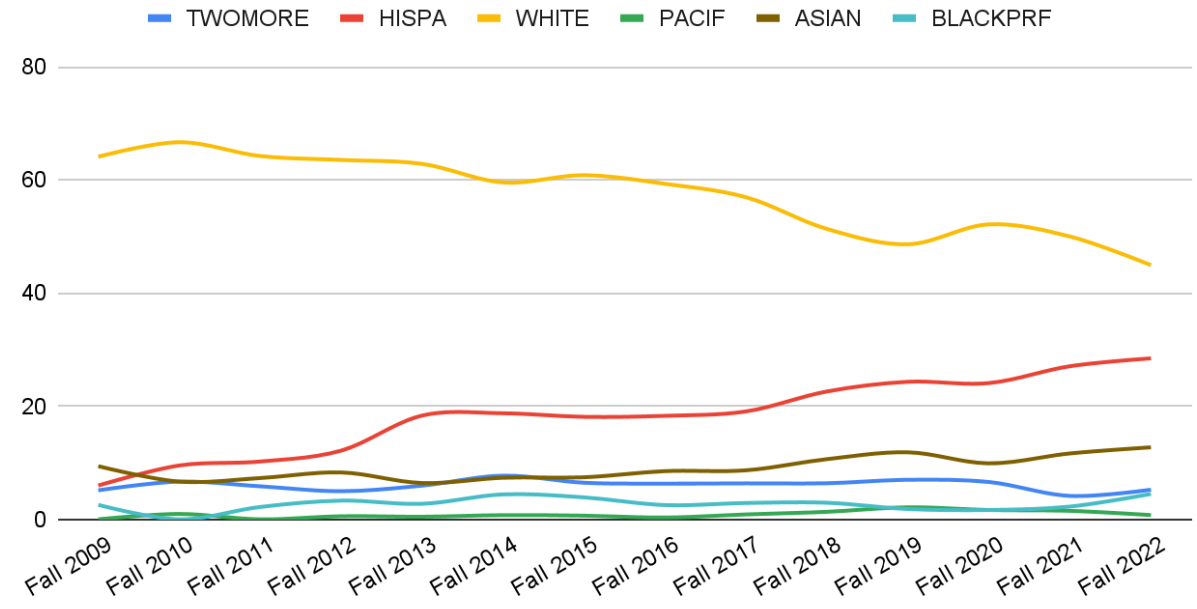

Figure 5.7 Ethnic breakdown of SSU CS majors (as % of total majors), Fall 2009-Fall 2022

Table 5.8 Data from 2021 for undergraduates in the SSU Computer Science program, SSU as a whole, and CS at North American research universities [Zweben22] as reported in the Taulbee survey. We have recalculated the data from Table B8 in the Taulbee survey to exclude “nonresident aliens” (12.3% of students at the reporting institutions), whose ethnicities are not further recorded.

	SSU Overall	SSU CS	Taulbee
Multiracial	9	4	4
Hispanic, any race	34	27	13
White	44	50	46
Alaska/Hawaii Native / Native American / Pacific Islander	2	3	0.3
Asian	6	12	30
Black	4	2	7
Unknown	2	2	

Strategies to support and recruit underrepresented minority students overlap heavily with those used to support low-income and first-generation students, which we discuss in the next section.

5.1.3 First Generation CS Majors

The number of first-generation CS majors follows the same diversity trends that the greater campus has experienced in the recent past. Figure 5.9 shows the steady rise of this demographic in our program.

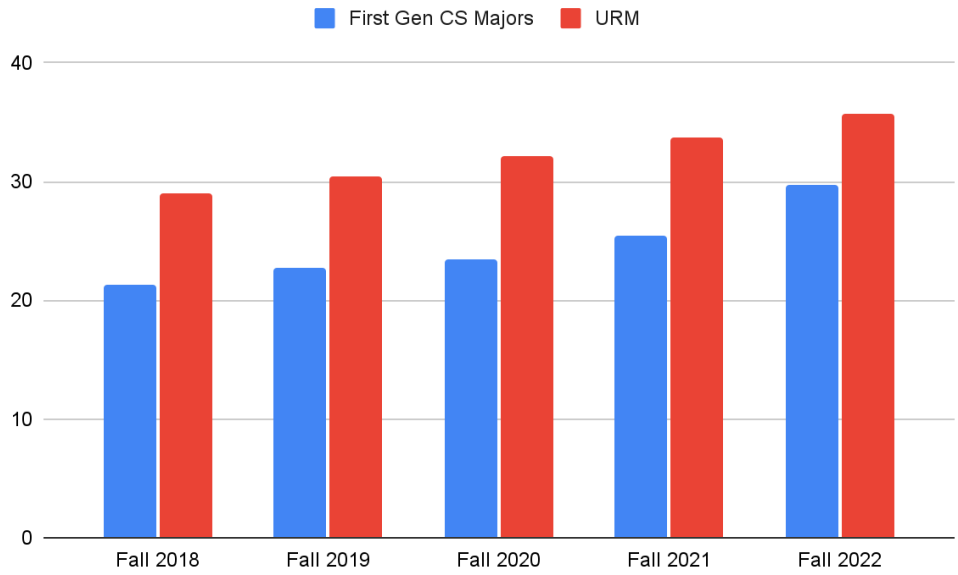

Figure 5.9. The rise of first-generation & URM computer science majors

5.1.4 Student Demographics: Age

An overwhelming majority of CS majors are between ages 18 and 24, as shown in Figure 5.10. Starting in Fall 2012, the percentage of CS majors in the 18-20 age group has dropped while the percentage of students in the 21-24 age group has increased. This change is consistent with the recent rise in the proportion of transfer students among our majors.

The percentage of students in the 25-29 age category has also risen sharply, and it is now almost double that of SSU overall (Table 5.11). Since the number of post-baccalaureate students has not increased significantly (Figure 5.3), this increase is also attributable to transfer students. It remains to be seen how much of this increase is a temporary artifact of the pandemic. On the other hand, CS – as a major with clear links to a career – has always attracted a disproportionate percentage of transfer students. Furthermore, many other universities' CS programs are impacted due to the nationwide capacity crisis (see Section 1.9).

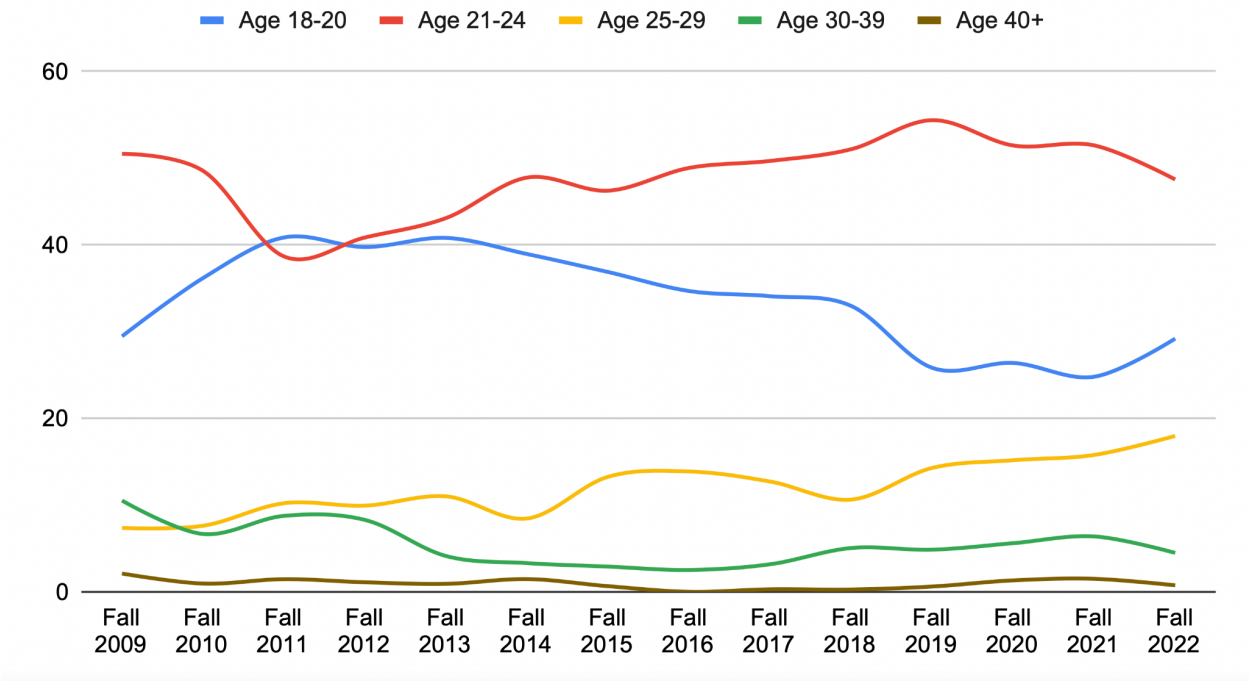

Figure 5.10 Ages of SSU CS majors (as % of total majors), Fall 2009-Fall 2022

Table 5.11 Age breakdown of overall SSU undergraduate population vs. CS majors, Fall 2022.

	SSU Overall	SSU CS
Age 18-20	27.2%	29.2%
Age 21-24	54.9%	47.8%
Age 25-29	10.5%	18.0%
Age 30-39	5.4%	4.5%
Age 40+	2.0%	0.8%

5.1.5 Unit Loads of CS Majors

The percentages of full- and part-time students in our program have changed little over time, averaging 75 and 25 percent, respectively, as shown in Figure 5.12. The percentage of students taking 15 or more units, as shown in Figure 5.13, fell precipitously during the pandemic. At the beginning of the pandemic, the percentage of students who were part-time increased. Since then, that percentage has returned to its normal baseline, while the percentage of students taking 13-14 units increased dramatically. It is not yet clear whether or not students will return to taking 15+ unit loads at pre-pandemic percentages, but this is a university-wide phenomenon.

The percentage of undergraduate students taking 15 or more units at SSU fell from a Fall 2019 high of 46.2% to a mere 25.84% in Fall 2022.

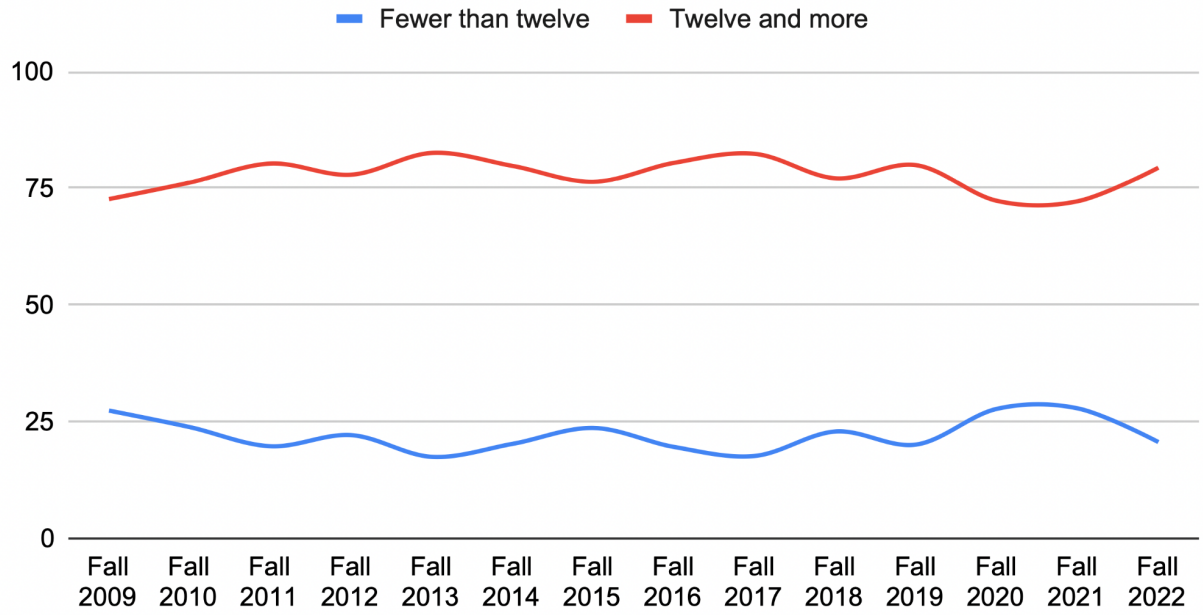

Figure 5.12 Percentages of SSU CS majors who are full-time vs. part-time (using the common 12-unit definition of full-time), Fall 2009-Fall 2022.

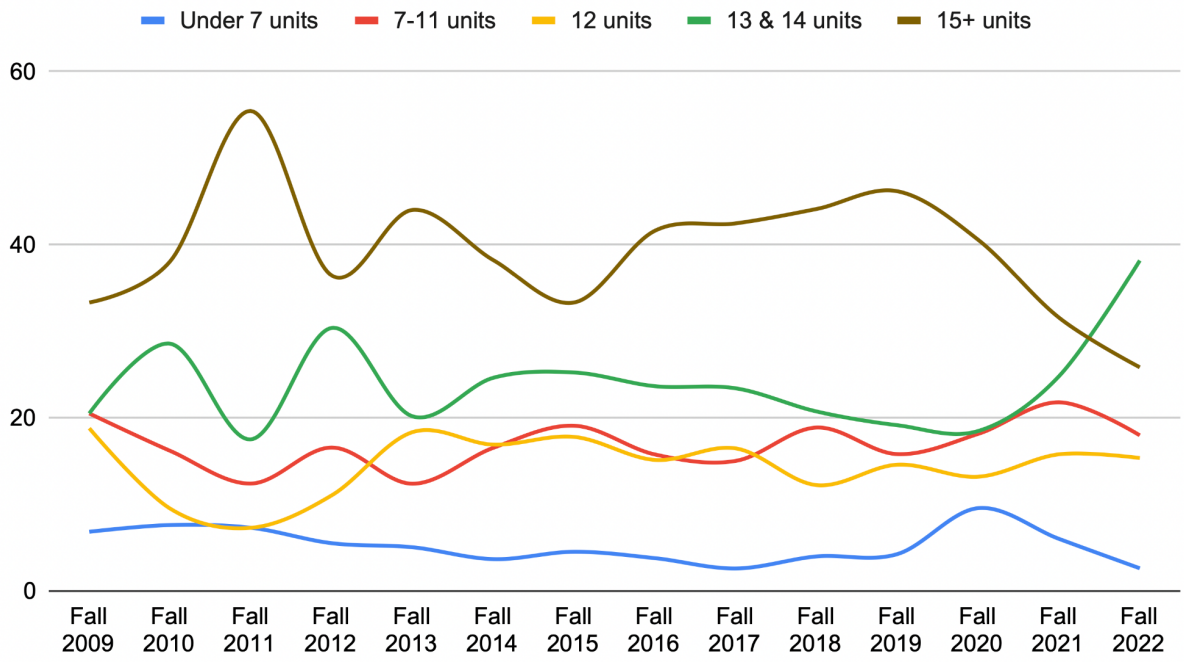

Figure 5.13 More detailed breakdown of SSU CS majors' unit counts, Fall 2009-Fall 2022

5.2 Student Demand

As Figure 5.14 demonstrates, the recent decline in the number of majors has been mainly due to the decline in the first-time freshman population, a trend that parallels that of the University. The number of matriculated transfer students, on the other hand, has trended upward since 2010. The result is that the transfer student population has been approximately equal to the first-time freshman population in three of the past four academic years, and significantly greater in the 2021-2022 year. Furthermore, because transfer students spend less time at SSU than first-time freshmen, their representation in our graduating classes have been even higher.

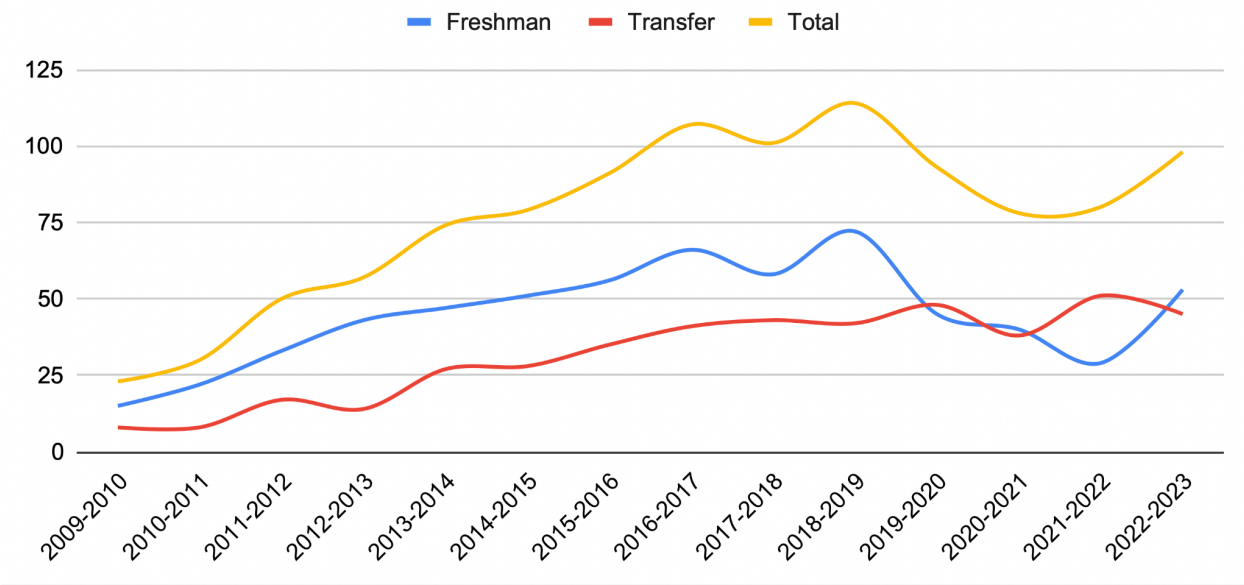

Figure 5.14 Newly matriculated freshman and transfer CS majors, 2009-2022

The upward trend in the number of incoming CS majors is reinforced by the number of new student applications for our program, shown in Figure 5.15. Note that this figure shows the applications *received* in a particular academic year, which correlate with first-time freshman matriculation the following year, and can correlate with transfer enrollment in either the same or the following year. Comparing both graphs shows that a trend in applications received one year is usually reflected in the number of newly enrolled students the following year, which bodes well for the 2023-2024 incoming class.

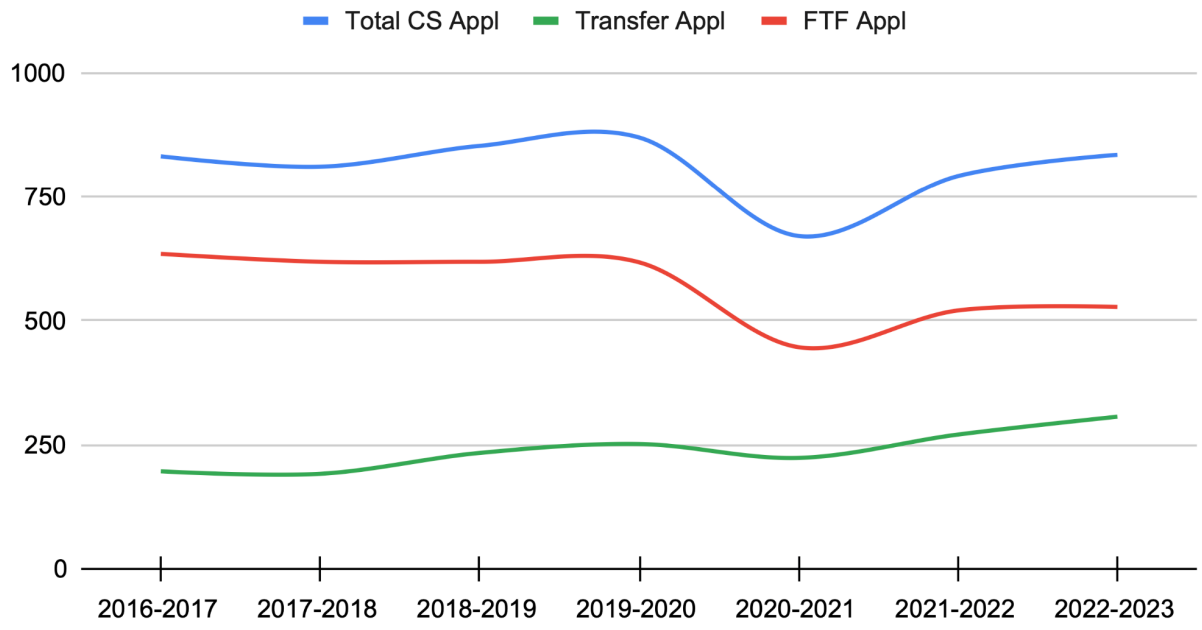

Figure 5.15 Applications received from prospective CS majors for the past 7 academic years

The rise in CS major enrollment, as discussed in Section 1, is a nationwide phenomenon that is fueled by the vibrant job market. The U.S. Bureau of Labor Statistics[3] projects the job market for *Software Developers, Quality Assurance Analysts, and Testers* to increase by 25% during 2021-2031 and categorizes this level of growth as "*Much faster than average.*" Many of our graduates accept jobs that fall into this BLS category. We expect student demand to remain above our capacity for the foreseeable future.

5.3 Retention and Graduation

We have provided comparisons of the Department's 4- and 6-year graduation rates for the first-time freshman and 2- and 4-year graduation rates for transfer students with those of the SSU and 6 other CSU campuses whose number of undergraduate population and their number of computer science majors straddle those of ours[4]. Additionally, we have made comparisons of the 4-year retention rates of our majors with those of the SSU for both, first-time freshmen and transfer students.

[3] https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm. Information Security Analysts grows at 35% (much faster than average), Computer Systems Analyst, and Database Administrators and Architects each grow at 9% (faster than average.)

[4] Monterey Bay is one of our comparable CSUs. However, the CSU dashboard did not have time-to-graduation data for this institution for several years and as a result, we excluded it from consideration.

The last year for which data for 4-year retention and graduation rates are available is Fall 2018, which puts graduation date at Spring 2022. For 6-year graduation rates, it is Fall 2016.

In order to contextualize the trends in retention and graduation rates that we have observed, Figures 5.16, 5.17, and 5.18 show the Department's FTES and FTEF for tenure and tenure-track faculty (we have truncated the dates to match those of the retention and graduation rates). We will reference these figures in the next two subsections.

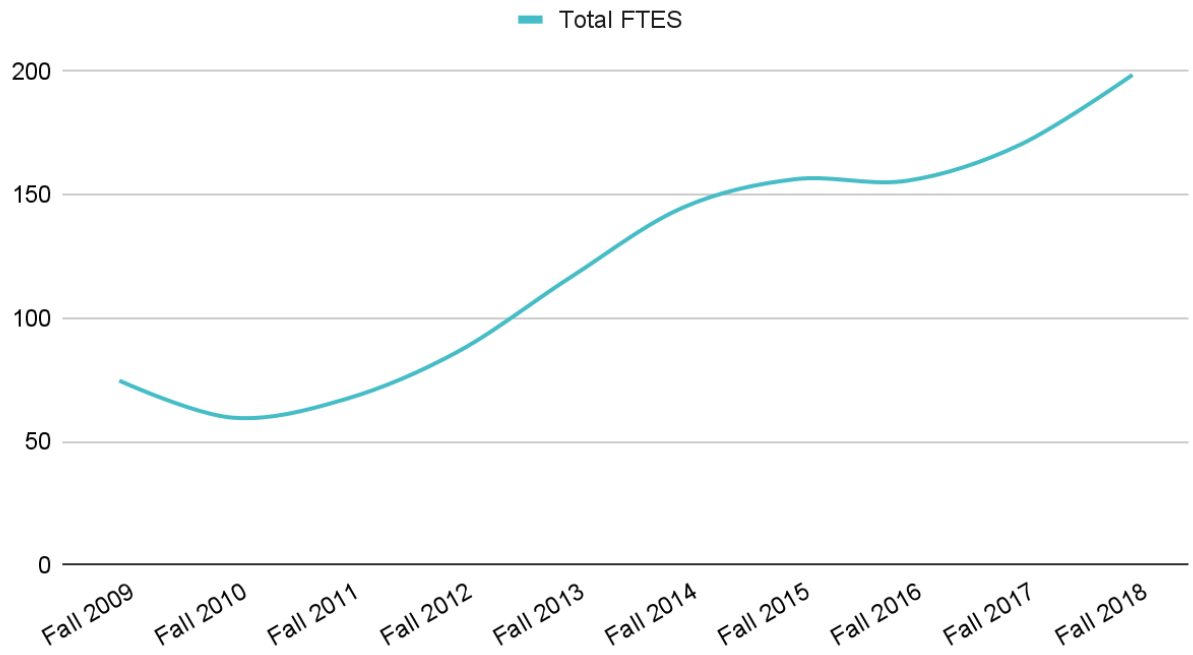

Figure 5.16 FTES trends for courses that are required for a BS in CS degree. The data stops in Fall 2018 to make it comparable to range of dates for 4-year retention and graduation rates

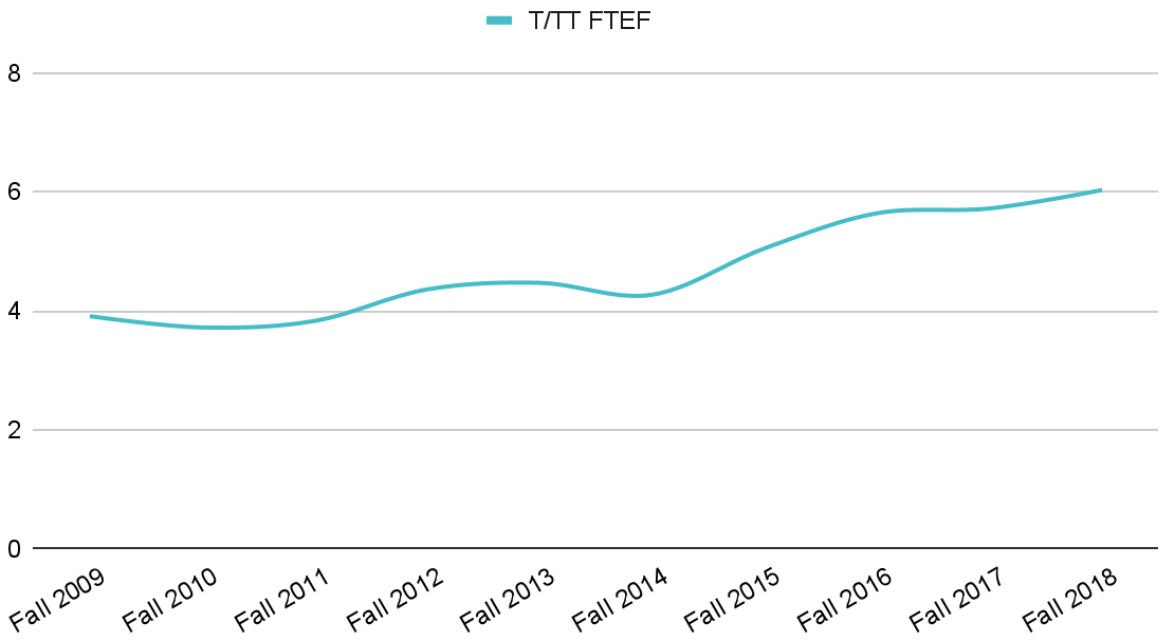

Figure 5.17 Full-time Equivalent Faculty trends for tenured and tenure-track faculty in the CS Department. The data stops in Fall 2018 to make it comparable to range of dates for 4-year retention and graduation rates

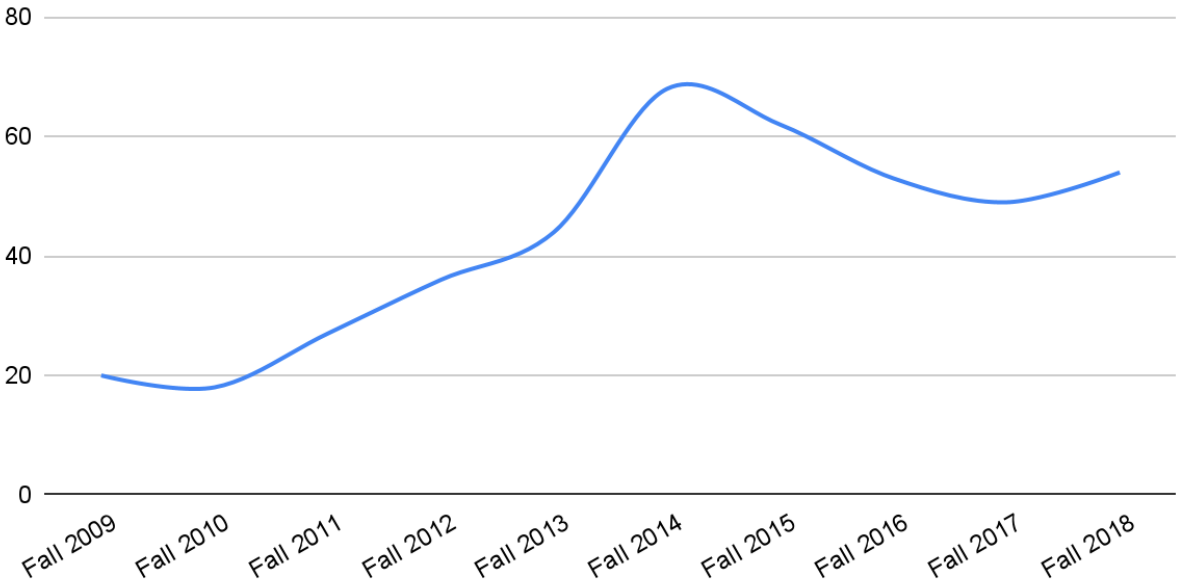

Figure 5.18 SFR for the CS Department. The data stops in Fall 2018 to make it comparable to range of dates for 4-year retention and graduation rates

5.3.1 Retention Trends[5]

The 4-year retention trends, as shown in Figure 5.19, in Computer Science for the most part closely track, and at times fare better than, that of the greater campus for first-time freshmen. For transfer students, as shown in Figure 5.20, the Department's retention rates are lower and reflect the stafficing and course offering challenges that we have experienced during this review period.

[5] In this section, we compare the department's workload metrics, such as FTES, for a given semester with the retention and years-to-graduation for the same semester. We understand that the added workload due to the rise in FTES in one semester doesn't immediately affect the retention rate in the same semester. But, the comparison is still sound as it demonstrates the cumulative effect of the rise in workload.

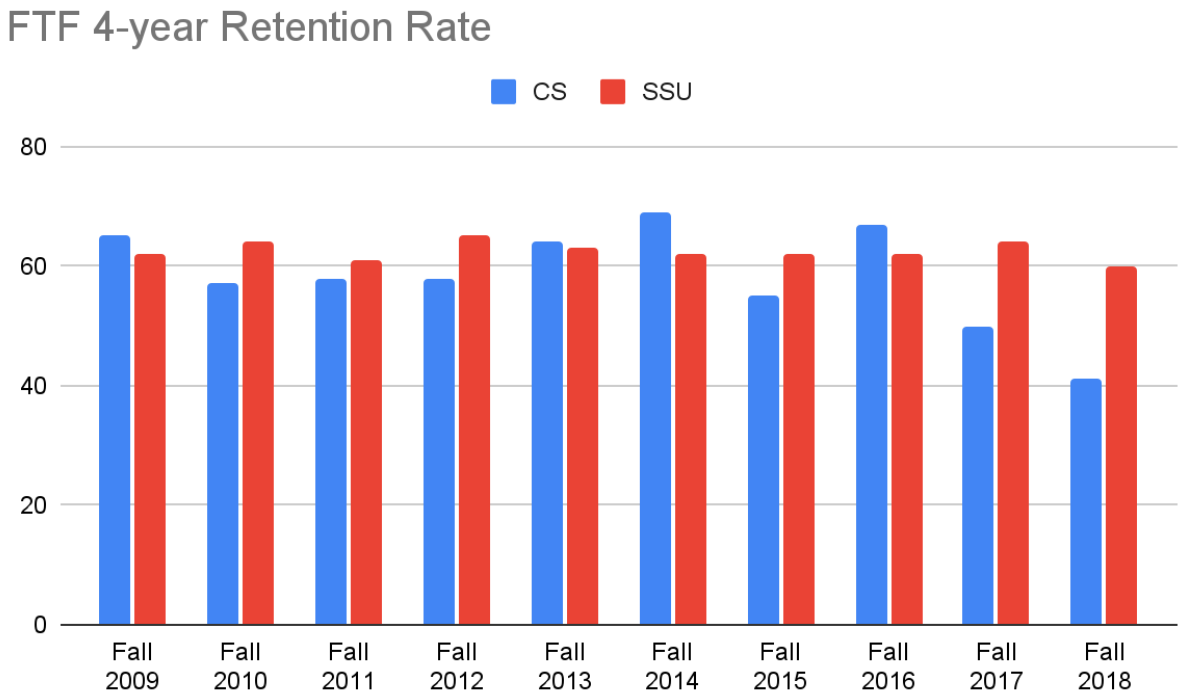

Figure 5.19 First-time Freshman 4-year retention rates for CS majors and those of the SSU

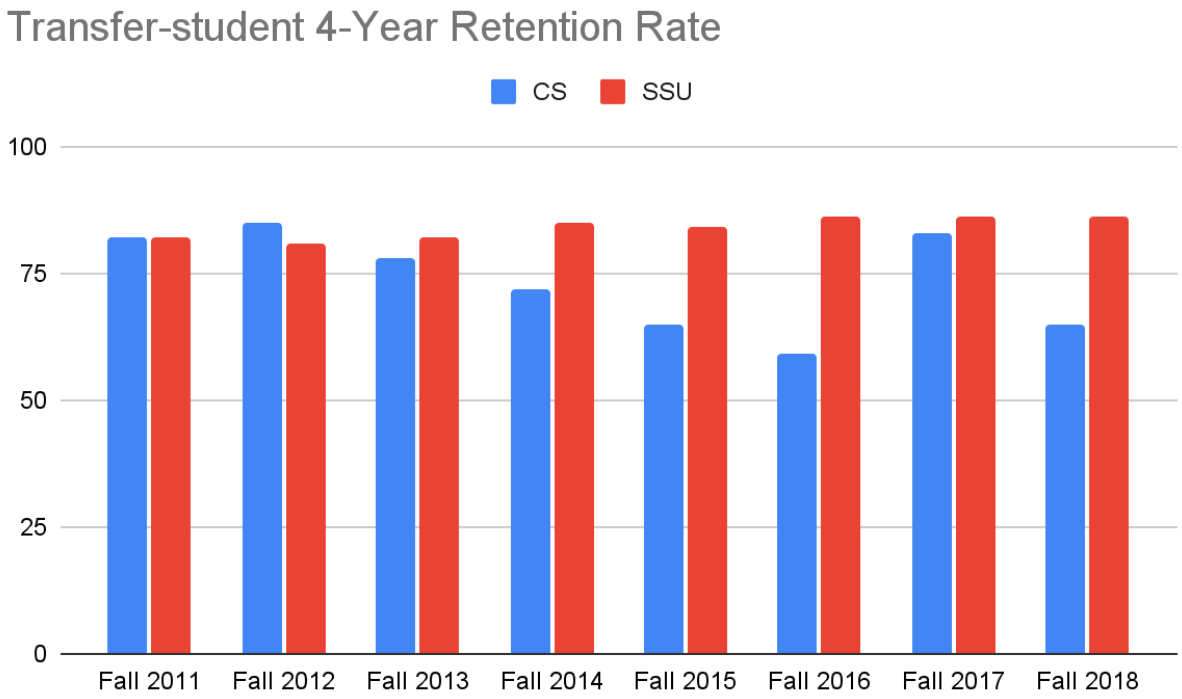

Figure 5.20 Transfer-student 4-year retention rates for CS majors and those of the SSU

A quick comparison in the rise of FTES and SFR (Figures 5.16 and 5.17) and relative flat nature of FTEF (Figure 5.18) provide context for the drop in the retention rates of Fall 2014, 2015, and 2016. Additionally, the drop in the retention rates in Fall 2017 and 2018 show the toll that the rise to 200 of the FTES takes on an understaffed department.

5.3.2 Time to Degree

The 4-year graduation rates for the first-time freshmen CS majors, as shown in Figure 5.21, from Fall 2009 to Fall 2017, either track or are better than those of the SSU's while the 6-year graduation rates are, shown in Figure 5.22, on the average, within 6 percentage points of those of the SSU's. These trends reveal that we are able to positively influence the educational trajectory of first-time freshmen through support systems such as advising and Club activities. The drop in the 4-year graduation in Fall 2017 and 2018 matches the steep rise in FTES (Figure 5.16) and high SFR (Figure 5.18). At that time, the department was over its capacity and as a result, we were not able to offer enough sections of our courses to support the rise in our number of majors.

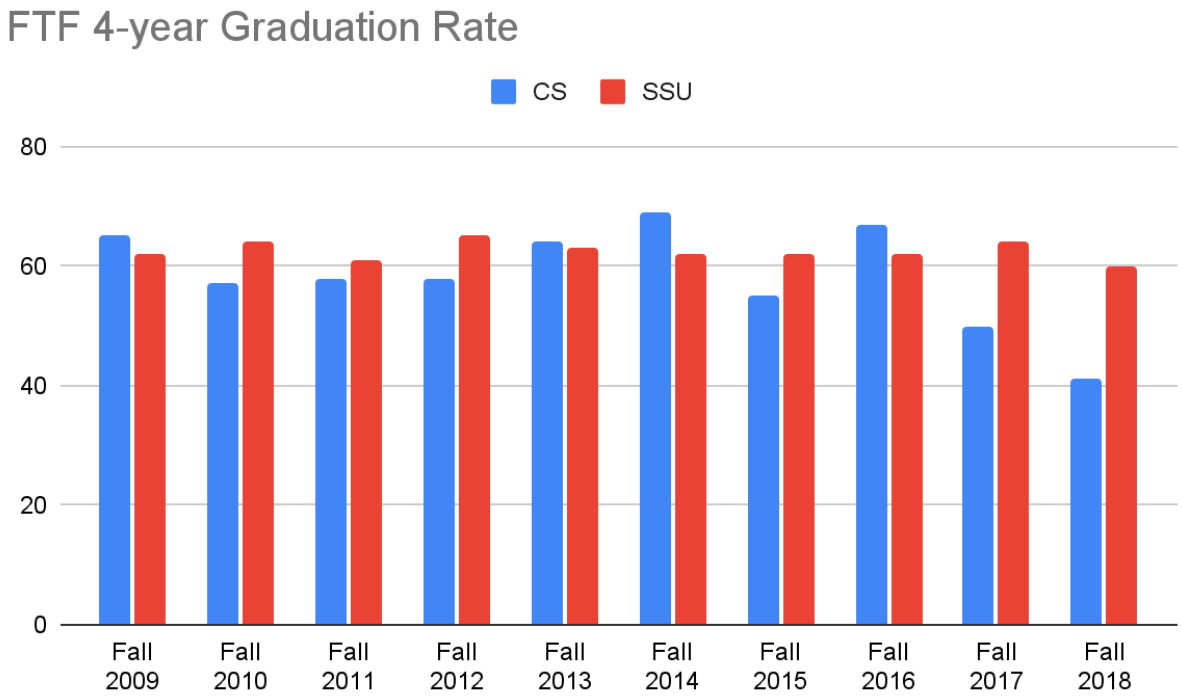

Figure 5.21 FTF 4-year graduation rates for CS majors and those of the SSU. Compare the drop in CS number in Fall 2017 and 2018 with steep rise in FTES of Figure 5.16, high SFR in Figure 5.18.

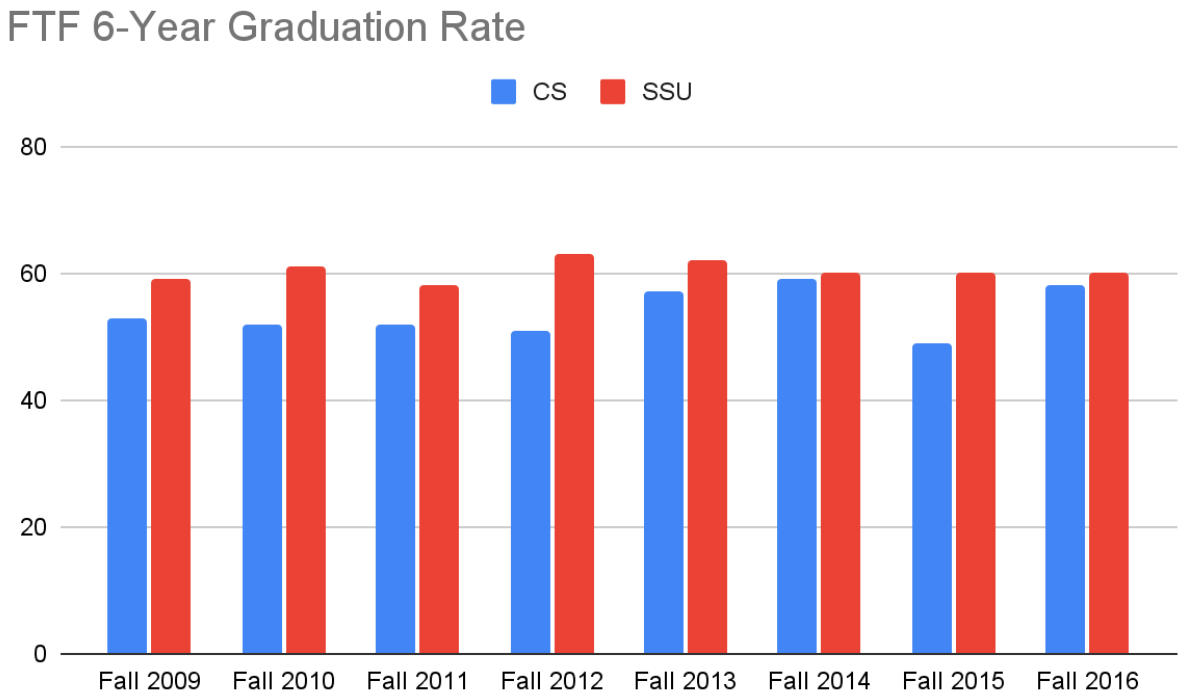

Figure 5.22. FTF 6-year graduation rates for CS majors and those of the SSU.

The graduation rates are even more impressive when we compare our program with those of computer science programs at *comparable* CSUs, shown in Figures 5.23 and 5.24 In a majority of cases, both in 4- and 6-year graduation rates, our program either performs better than all comparable programs or it is only below one or two other programs. In a few cases, it it is in the middle of the pack. Never even close to the bottom.

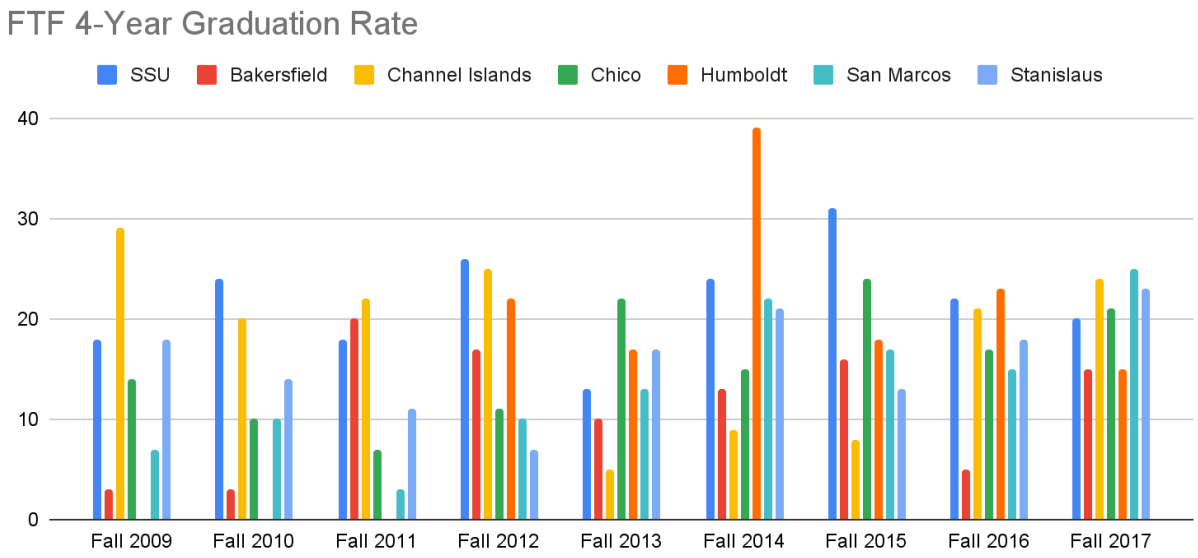

Figure 5.23 First-time freshman 4-year graduation rate for SSU and 6 other CSU computer science departments that are comparable to SSU's

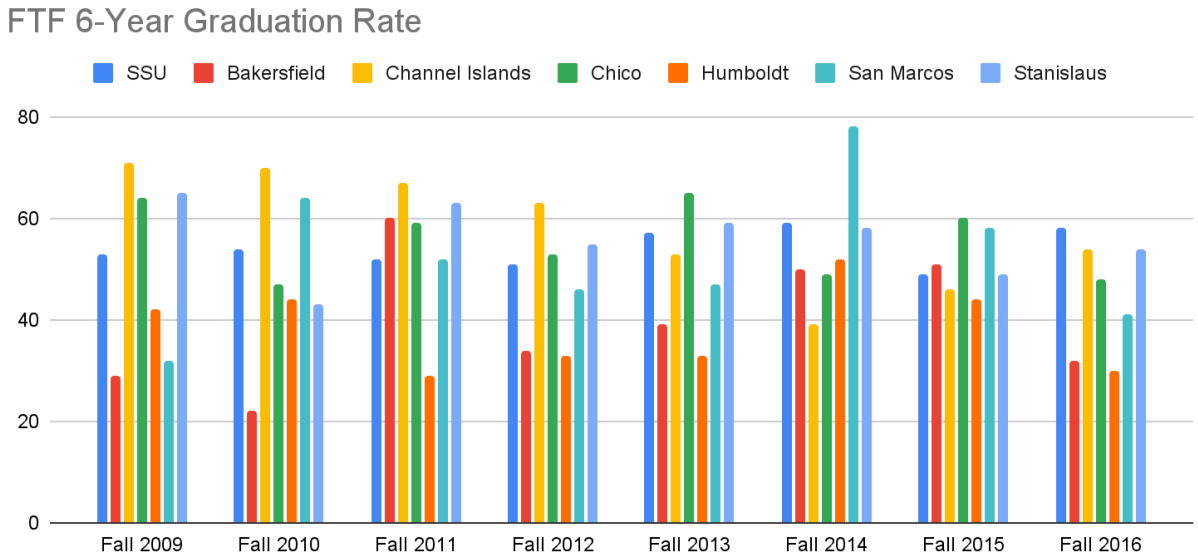

Figure 5.24 First-time freshman 6-year graduation rate for SSU and 6 other CSU computer science departments that are comparable to SSU's

As shown in Figure 5.25, the 2-year graduation rates for the transfer-student computer science majors are lower than those of the SSU's. Computer science is a vertical major and as such, even those students who are qualified to take CS 315 when they enter our program, have to

take every required course at the right time, starting with their first semester, to graduate in two years. When seats in courses are scarce and they build long waitlists, which happened many times between Fall 2014 and Fall 2020, in spite of the fact that we set aside some seats for transfer students, it has not been easy for transfer students to get all courses that they need in their first semester at SSU.

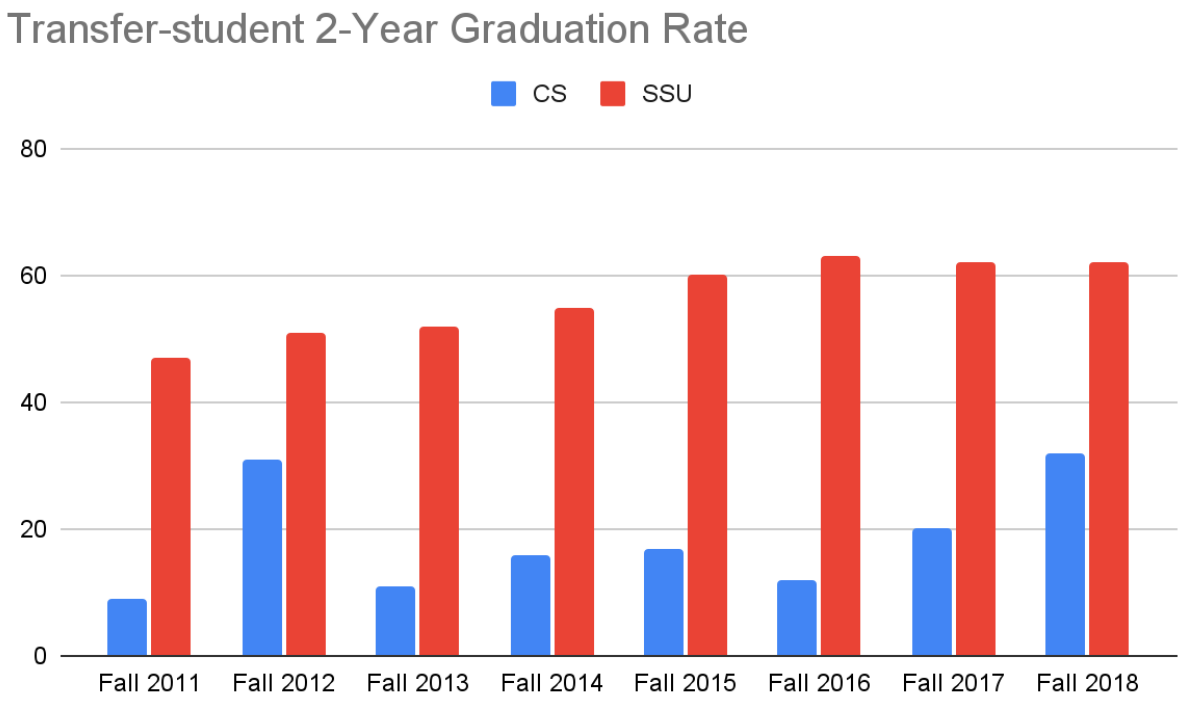

Figure 5.25 2-year graduation rates for CS majors as compared to those of SSU.

The graduation rates for transfer CS majors is much closer to those of the SSU as shown in Figure 5.26. This Figure is similar to that of Figure 5.20, which shows the 4-year retention rates for CS majors. The inability for transfer students to graduate in 4 years, which was the case in 4 of the five years starting with Fall 2014, resulted in students dropping out. The reader is encouraged to compare the rise of FTES and SFR of Figures 5.16 and 5.17, the relative flat nature of FTEF in Figure 5.18, and the drop in the 4-year graduation rate of transfer CS majors.

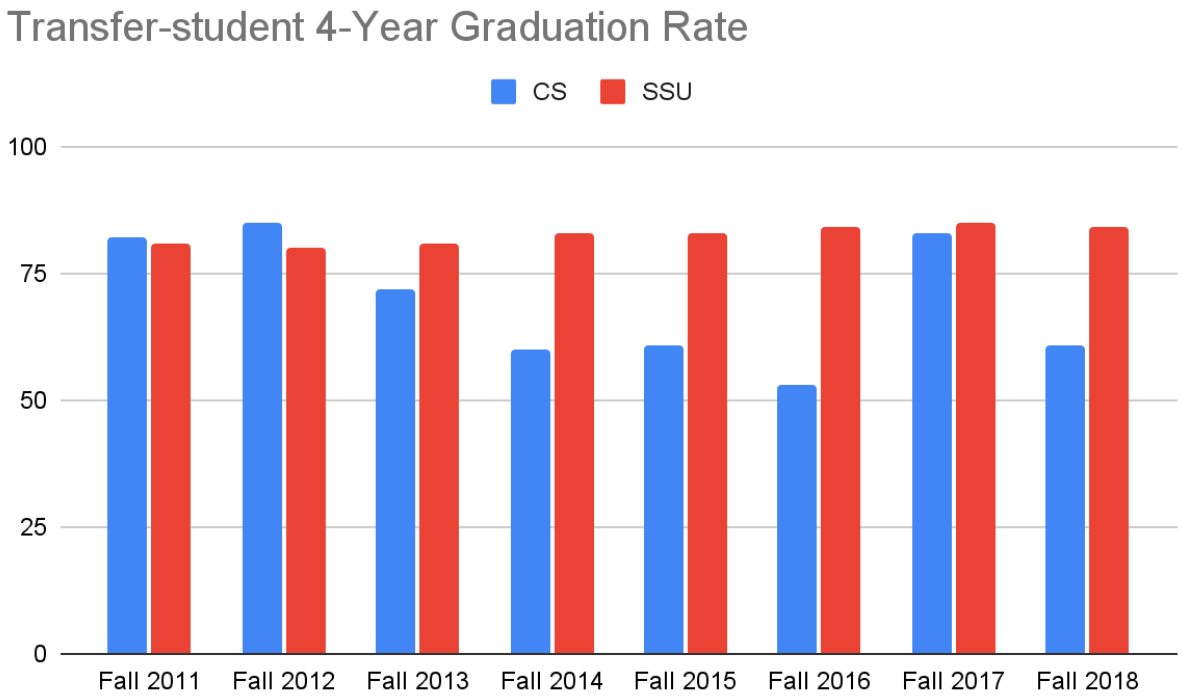

Figure 5.26 4-year graduation rates for CS majors as compared to those of SSU.

Figures 5.27 and 5.28 below show the 2- and 4-year graduation rates for our program and those of 6 comparable CSU computer science programs. Once again, our program has a much higher graduation rates than the majority of the other CSU programs.

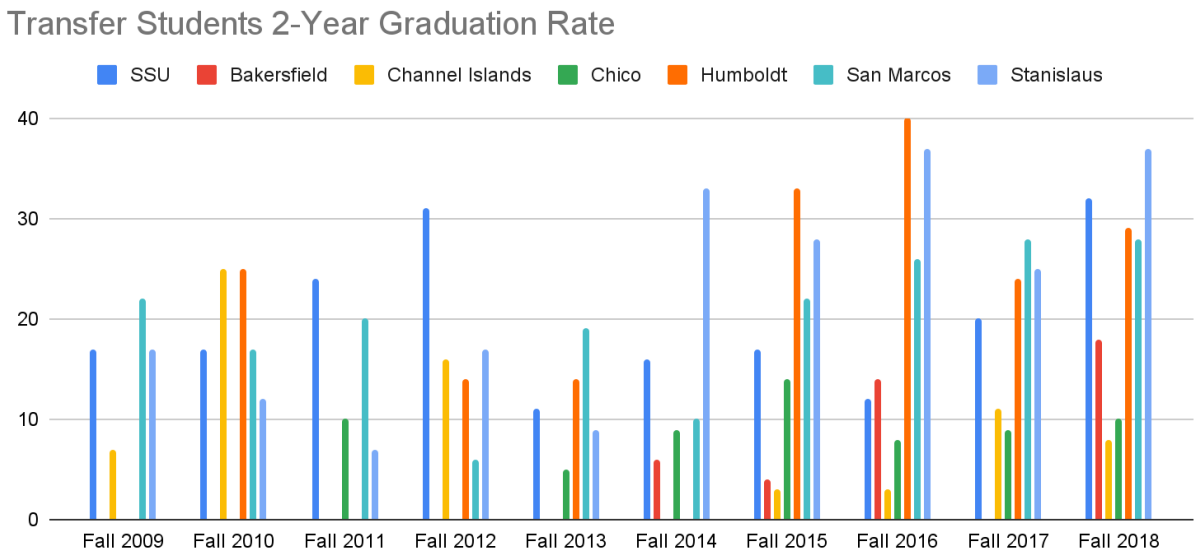

Figure 5.27 Transfer-student 2-year graduation rate for SSU and 6 other CSU computer science departments that are comparable to SSU's. The missing bars indicate a zero value

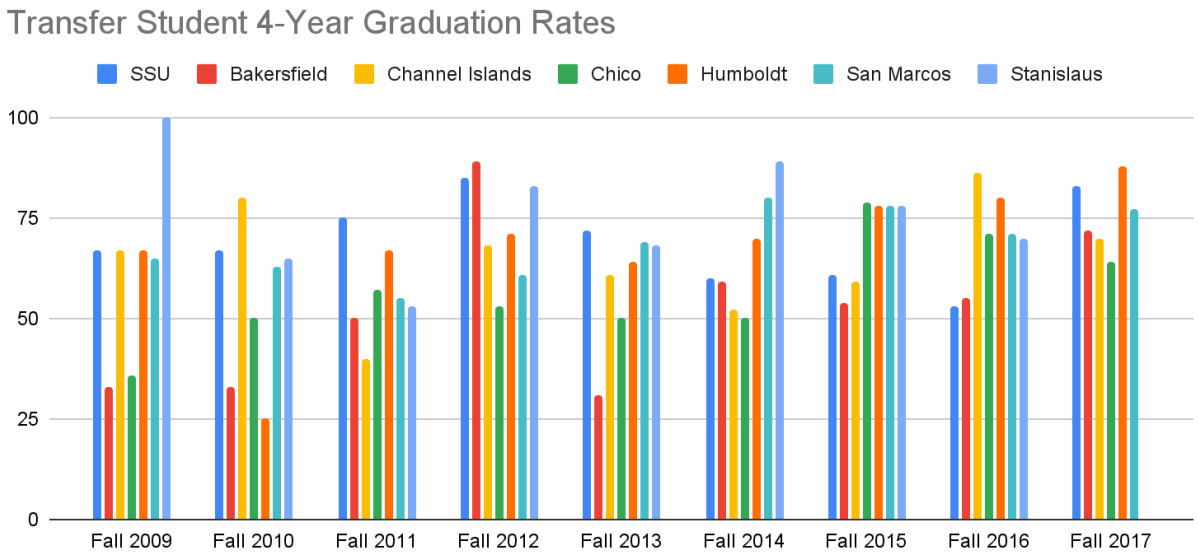

Figure 5.28 Transfer-student 4-year graduation rate for SSU and 6 other CSU computer science departments that are comparable to SSU's

5.3.3 Degrees conferred

The following two charts show the number/gender of our graduates for this review cycle. The steady rise of the number of graduates from our program for a large majority of academic years during this cycle is just impressive.

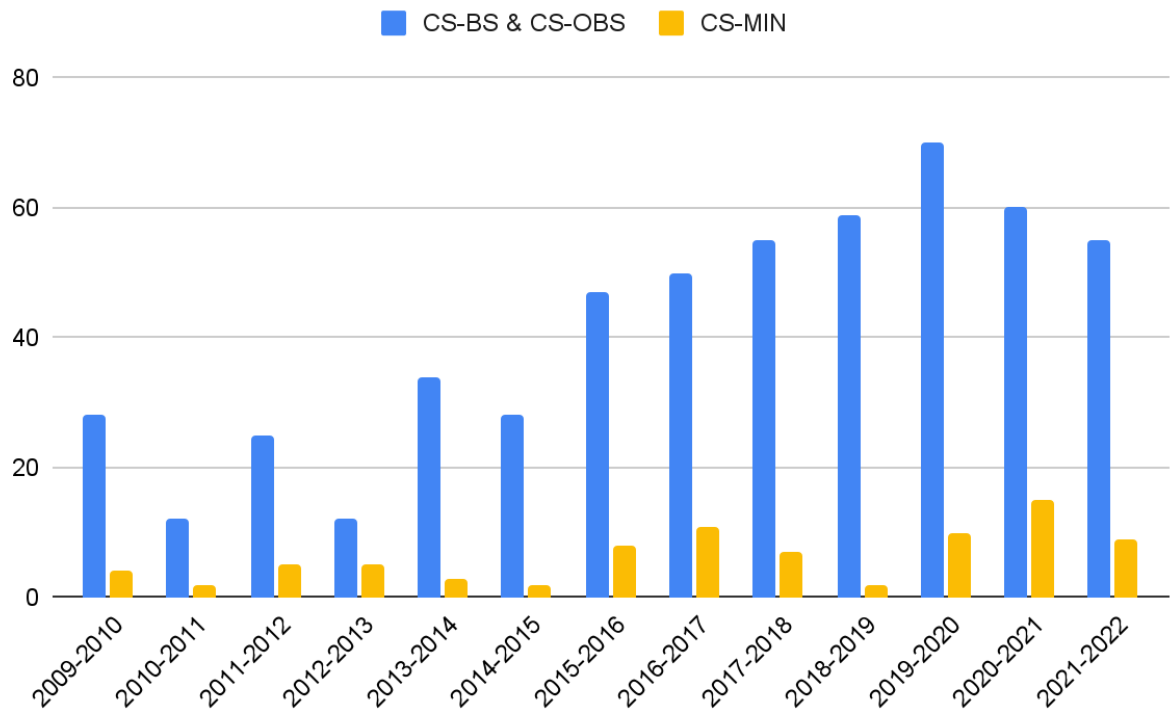

Figure 5.28 Number of graduates with a BS in CS and with a Minor in CS

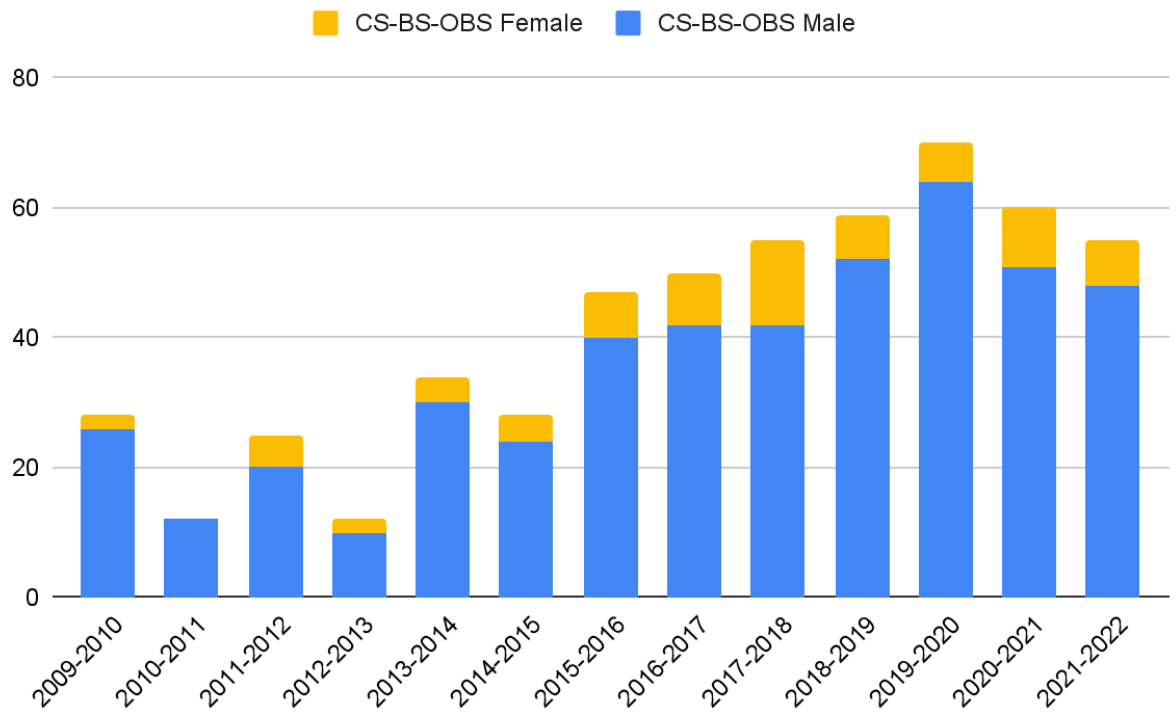

Figure 5.29 Number of graduates with a BS in CS by gender

5.3.4 DWF Rates

The DWF rates for three core Computer Science courses -- Programming I (CS 115), Programming II (CS 215), and Data Structures (CS 315) – are high. They are in the company of 20 courses offered by the departments in the School of Science and Technology, and 39 courses offered by all departments at SSU, whose DWF rates are higher than 25 percent when calculated for the academic years 2012 through 2021. Figures 5.30 and 5.31 show these three courses among the other high-DWF courses in SST (Figure 5.30) and the University overall (Figure 5.31). Figure 5.32 shows the DWF rates for all computer science courses for the 2012 through 2021 academic years.

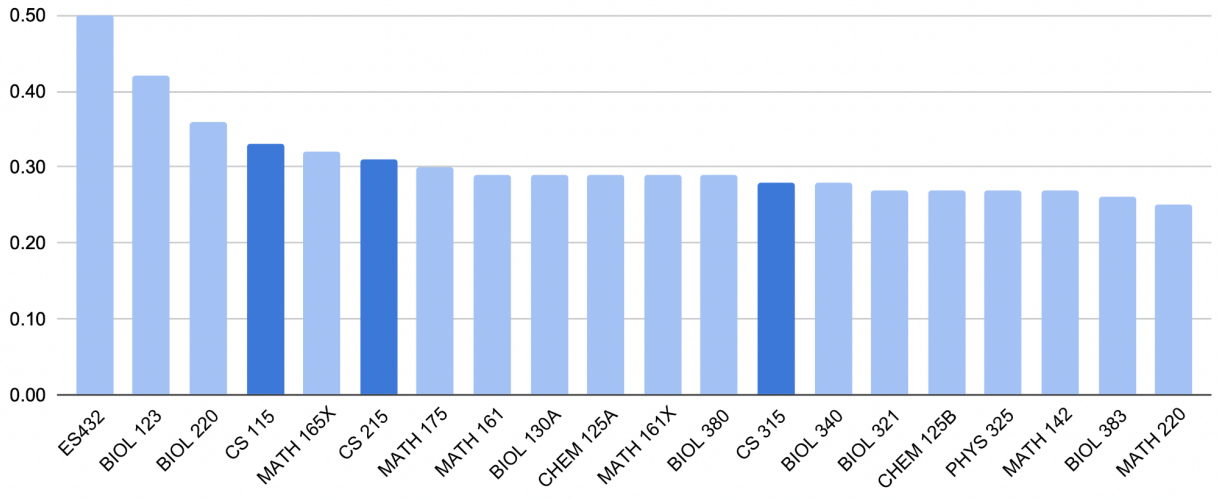

Figure 5.30 SST courses with DWF rates of 25% and higher during the 2012-2021 academic years

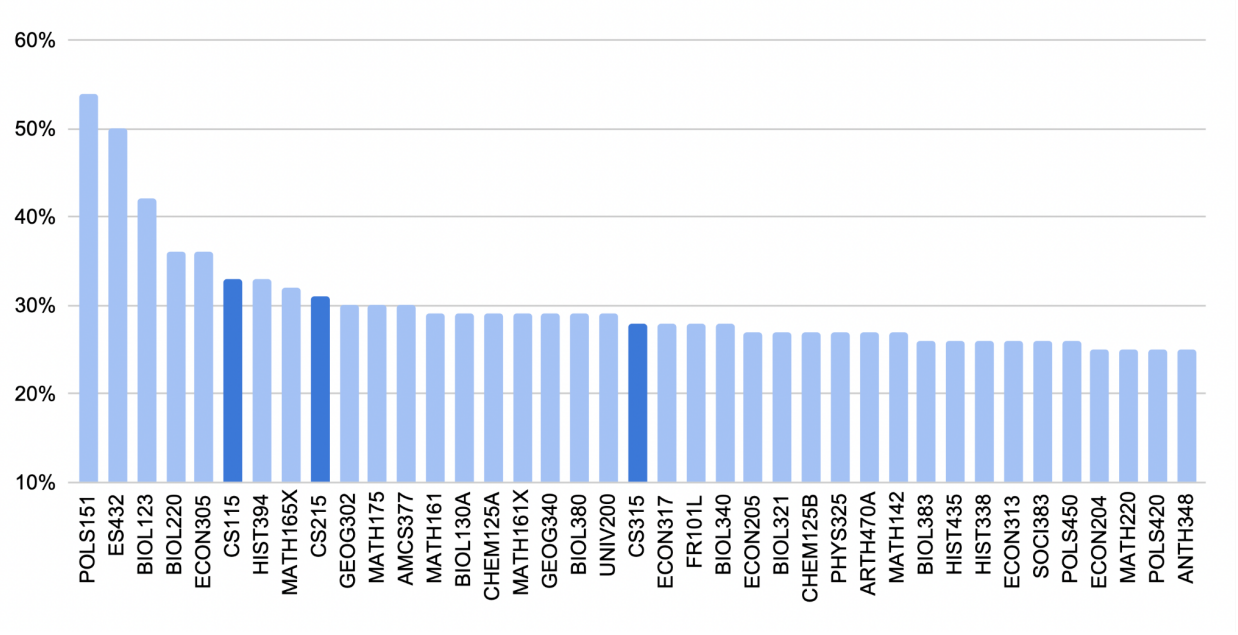

Figure 5.31 All SSU courses with DWF rates of 25% and higher during the 2012-2021 academic years

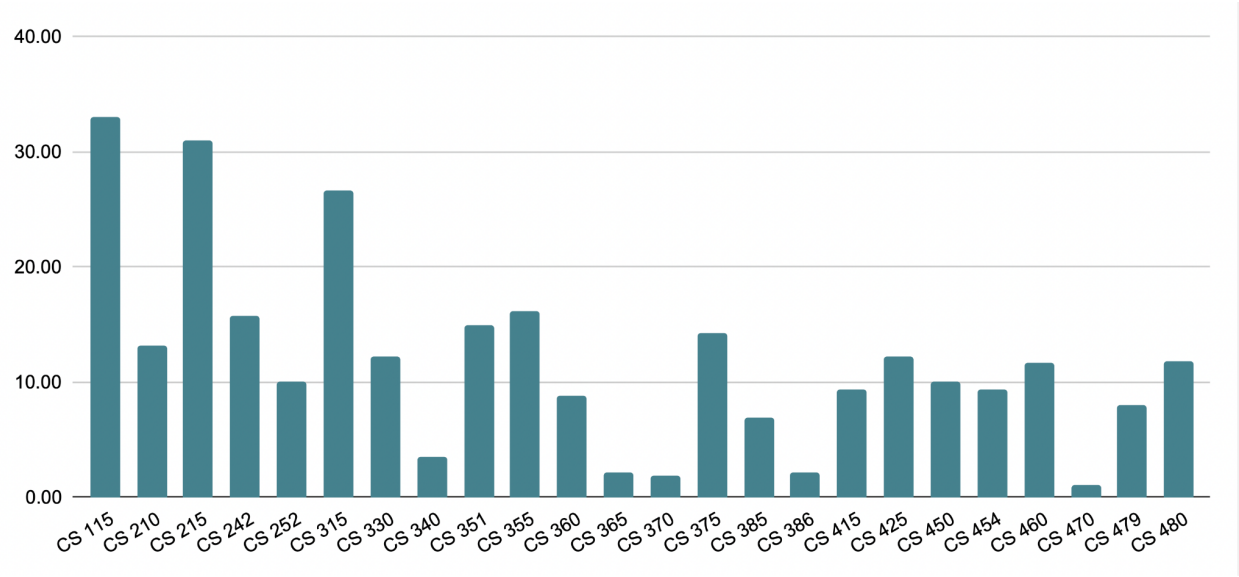

Figure 5.32 DWF rates for all CS core and major elective courses during the 2012-2021 academic years

To an extent, these high failure rates reflect a departmental commitment to ensuring that students only pass these fundamental courses if they have demonstrated the skills and abilities needed to succeed in subsequent courses. By that measure, we have been fairly successful – CS 355 (Database Management System Design) has the next highest failure rate at 16 percent, 10 full percentage points lower than CS 315 at 26 percent. The overall DWF rate for courses in the School of Science and Technology in Fall 2022 was 12 percent, including GE courses, with 80 SST courses having failure rates higher than CS 355's 16 percent.

We would, of course, prefer to have more students succeed in their first attempt at the CS 115-215-315 sequence. Drs. Mark Gondree and Gurman Gill, who teach CS 115 and 215 most often, are engaged in professional development around DWF rates specifically (through SSU's Academic Programs office) and equitably serving Hispanic students (through the TIPS grant). However, it is unlikely that pedagogical improvements will fully solve this problem.

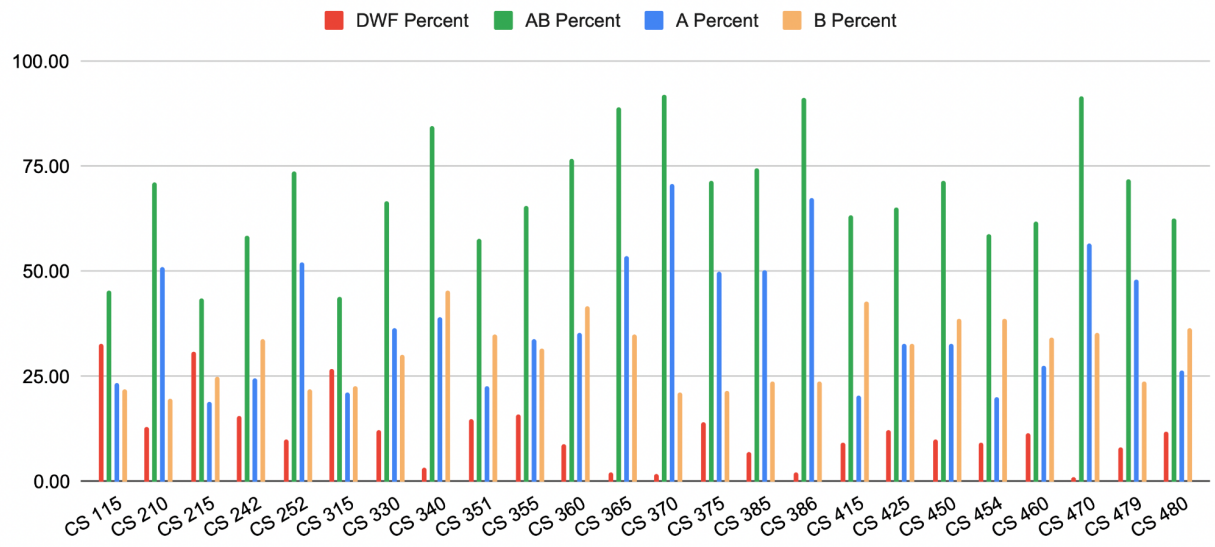

Figure 5.33 DWF alongside AB (A, A-, B+, B, B-), A, and B rates for all CS courses, academic years 2012-2021.

High failure rates in programming courses – accompanied by high rates of A and B grades – are widely observed and remarked upon across the computer science education community and are consistent with what we see in CS 115-215-315, as shown in Figure 5.33. Studies investigating DWF rates among CS1 courses routinely observe failure rates between 28 and 33% [Bennedsen19]. Researchers have put forth numerous hypotheses, none universally accepted, for these bifurcated student outcomes [Patitsas16]. The ones that most apply to our student population are

1. **Differences in preparation.** The most salient differences are not necessarily in direct prior knowledge of computer programming, but rather in students' expectations for the major and their coursework. Often, students do not realize the extent to which problem-solving skills and creativity permeate a computer science education, starting with CS 115.
2. **Differences in time spent on coursework.** CS 115-215-315 require engineering effort on projects as well as time spent mastering the fundamental skills of programming. We recognize that students' ability to spend time on coursework is inequitably distributed and is affected by their work schedules, physical and mental health, and family circumstances. The high advising ratio in our department, and the rushed nature of orientation and transfer advising in particular, mean that we lack the resources to work with every student on calibrating a course load that is right for them.
3. **The holistic nature of assessment in programming courses.** Creating a working computer program requires the integration of numerous individual concepts. It takes a great deal of focus and persistence to discover ways to turn the statement of a problem into a computer program. The learning curve is steep and remains that way in each of the three programming courses, even for students who are academically strong.

We also face a fourth, more specific issue due to the intersection between California's modular, transfer-friendly system and the nature of programming curricula. As discussed in Section 5.2, our upper-division courses are dominated by transfer students who come to us from community colleges throughout California. We accept all courses designated as matching the lower-division California model transfer CS curriculum (C-ID courses), as well as having specific articulation agreements with other community colleges. However, the model transfer curriculum only lists concepts to be covered – not the languages, development tools, or overall level of independence and academic maturity expected of students. Inevitably, some students will find mismatches between their prior preparation and the courses they place into SSU.

Time to graduation is understandably a priority for transfer students, but even those who are willing to retake a foundational course are not always able to do so: our courses are often full by the beginning of transfer registration, let alone by the start of the semester when students may realize they are in over their heads. On the rare occasion that we have the staffing flexibility to offer CS 115, 215, and 315 at the same time, plus available seats in those courses at the beginning of the semester, we have seen students self-select from (for example) CS 315 into CS 215, with great success. We have often discussed the idea of a department-specific assessment for placement of transfer students into the 115-215-315 sequence, but we lack the resources to develop, administer, and validate such an assessment.

5.3.5 Challenges of teaching combined courses

As we mentioned in Assessment, CS 115, 215, and 315 are project-intensive combined courses -- each course consists of a combination of lecture and closed labs. When student demand is high, which has been the case for the majority of the semesters during this review cycle, two lab sections get combined into a single lecture, resulting in lectures that consist of around 50 students -- doubling the CSU's population-size recommendation based on the CS number of the course. Additionally, due to staffing shortage, the teaching of the labs are usually delegated to part-time instructors who inevitably have day jobs and are unable to attend lectures. Therefore, in spite of their best efforts, the lab instructors lack the context in which the material that is the subject of the lab, has been presented to the students during the lecture.

Moreover, this method of offering lab-based courses drastically increases the workload of the lecture-instructor, who is responsible for the cohesion of the course and design of all aspects of the course, including the labs. The result is complicated and time-consuming coordination logistics, exasperating the DWF rates of these courses, and leading to burn out. It is notable that due to the critical nature of these courses in students' computer science education, we make every effort to schedule the tenured and tenure-track faculty to teach the lectures of these courses.

5.4 Alumni Feedback and Outcomes

In Fall 2022, we invited the members of our LinkedIn alumni group to participate in an anonymous survey about their lives after graduation and their feedback about our program. Table 5.34 shows demographic and employment information about the respondents.

Table 5.34 Demographic and employment information from 2022 alumni survey

Employment Status	Income	Grad Year	Age Group	Ethnicity	Gender
Fulltime	More than $150,000	2010	35 - 39	Caucasian or White	Male
Fulltime	$80,000 -- $99,999	2021	20 - 24	Caucasian or White	Male
Fulltime	$100,000 -- $150,000	2019	30 - 34	Caucasian or White	Male
Fulltime	$60,000 – $79,999	2013	30–34	Mexican, Hispanic, or Latino	Male
Fulltime	$100,000 – $150,000	2021	25–29	Caucasian or White	Male
Graduate/professional school	$40,000 – $59,666	2020	20–24	Caucasian or White	Male
Fulltime	$80,000 – $99,999	2020	20–24	Caucasian or White	Male
Fulltime	Prefer not to answer	2022	30–34	Caucasian or White	N/A
Not employed	N/A	2021	20–24	Mexican, Hispanic, or Latino	Male
Fulltime	$100,000 – $150,000	2022	20–24	Caucasian or White	Male
Fulltime	$100,000 – $150,000	2019	25–29	Caucasian or White	Male

Table 5.35 summarizes the quantitative feedback about their experiences as SSU CS students. The number of respondents for each question was at least 12.

Question	Avg Score (max 5.0)
Satisfaction with the major	4.30

Scholarly and professional competency of the faculty	4.35
The quality of instruction you received in your computer science courses	4.21
Faculty awareness of new developments in Computer Science	3.95
Support and concern as individuals by CS faculty and staff	4.35
Support and concern for academic success from CS faculty and staff	4.25
Variety of elective courses offered	3.2
Frequency with which elective courses were offered	2.95
Frequency with which required courses were offered	3.60
Quality of campus advising resources on campus	3.82
Quality of departmental advising from the CS program	3.85
The intellectual challenge you received in the major	4.15

The following is a selection of responses to the question: “Tell us about one of your positive experiences as a CS major”

1. The faculty was instrumental for my growing passion in CS. I went from wanting to do it because "it was good paying" to wanting to explore, learn, and understand CS more than others.
2. I really enjoyed working on algorithm projects with a partner. I would end up writing them faster than expected, then apologizing to my partner for finishing the project so quickly. He would then point out some efficiency issues or edge cases with my code, and we would have to revise it.
3. My capstone course (team-based programming project with Dr. XXX [edited out intentionally]) was the most essential course in my major. This course required teamwork, a consistent application of all SWE skills learned in the major, and the ability to learn new skills quickly. This is the course that has helped me the most at my current job and also gave the best discussion points during my interview.
4. Alternatively, there is my entire experience with undergraduate research w/ Dr XXX [name edited out intentionally]l. He was an excellent mentor who started me down the path that led to a publication in Audio source separation enhancement and a job at Google.
5. Individual attention due to small class sizes. A lifetime mentor….

As the responses indicate, the alumni are happy with the rigor of their CS education and hold the faculty in high regard as it relates to their competency and respecting the students as individuals.

The greatest source of dissatisfaction in this survey was in our elective offerings. For context, our program does not have tracks or concentrations. Rather, it provides a broad and in-depth foundation that prepares students for an exceptionally diverse job market and graduate programs. However, we do provide a narrow level of specialization by cycling through elective courses, usually offering two to three 3-unit electives each semester. Students generally take two 3-unit CS electives over the course of their college careers.

Our elective offerings each semester are heavily influenced by the availability of the faculty and their areas of specialty. When student-faculty ratios are high, which has been the case for the last decade, the number of seats in elective courses is far fewer than the number of students who need to take them, and it is generally easier to find outside experts to teach electives in their areas of expertise than to teach our core courses. This can result in a mismatch between student demand and faculty supply in elective offerings.

The information in Table 5.35 above is consistent with our general sense of student outcomes at graduation. Computer science, somewhat unusually for the sciences, does not require a graduate degree for meaningful and well-compensated work in the field, and the vast majority of our students understandably opt to go straight into industry. Our goal is to ensure that students have the information and academic preparation to enter graduate programs if they choose to do so, while preparing the median student for direct entry into the tech workplace.

6. Reflection and Plan of Action

The faculty of the CS department are passionate and active members of the Sonoma State community, invested in mentorship following the teacher-scholar model and student success. While the department has seen tremendous growth in its majors, it has also suffered (along with the university) from common challenges, including fires, pandemics, declining FTF enrollment. Our self-study has revealed questions to address and opportunities to act on, and we reflect on several themes below. We look forward to the program review's recommendations.

6.1 Retain a Diverse Faculty Body

Our recent alumni survey reflects the appreciation of our students toward those of our adjunct faculty with current positions in industry, whom we have largely recruited from among our own alumni. Contact with industry professionals clearly enhances our students' learning experiences. It is also apparent that local industry professionals are largely unswayed by the opportunity for full-time lecturer positions at Sonoma State. Anecdotally, their motivation for serving as adjuncts appears to be anchored in the value of their alma mater, the pleasure of interacting with former professors, and the appreciation earned from students and faculty. The challenges of retaining qualified, interested adjuncts from industry (to the benefit of our students) seem gravely under-appreciated by administration and staff. So too is our (in)ability to "lean more heavily" on temporary faculty in times of need — when permanent faculty earn release time associated with service or make use of contract benefits including sabbatical leave, DIP, and paternity leave. Growing and diversifying our adjunct pool is expected to continue as a challenge. Partnerships with staff and administrators may be possible with the shared goal to shield our valuable (almost volunteer) adjuncts from any bureaucratic stressors and reduce administrative barriers challenging them.

The challenges in attracting and retaining adjunct faculty are punctuated by the recent loss of all CS tenure-track faculty within a 2-year period. The sudden departure of our tenure track faculty demands investigation. The events call into the question the ability of the University to support and retain minority faculty in technology fields. In particular, all CS faculty resignations were faculty who are women and/or BIPOC. The Electrical Engineering department is our peer technology department in SST. In their nearly 20-year history, they have never had any female tenured or tenure-track faculty. SST recently terminated the interdepartmental "Women in Tech" group that served CS, EE, and physics. We question the strength of the school's record in retaining minority faculty in technology fields during any period of duress. These individuals are, bluntly, people with options who won't put up with nonsense.

While the department can make clear business arguments for hiring new faculty to re-build in the face of losses and support near-certain major growth, the department is wary of re-building. Losing faculty is both disruptive and costly. The retention of women/BIPOC faculty in technology fields at Sonoma State warrants serious investigation (including administrative barriers and conflict resolution processes). It is not appropriate for that investigation to be conducted under

the auspices of this self-study. It may be inappropriate for the results of any such investigation to originate from the CS department, itself.

6.2 Continue to Support our Diverse Student Body

Students strongly value our program's technical rigor and career preparation, as reflected in our Alumni survey. The overwhelming majority stated that their CS degree had been essential to their first job and that the program was important to their overall career. From this feedback, we believe our alumni feel well-served by the technical quality of our program. Since the period of the last review, we have introduced some career-related curricular enhancements, like CS 391. Our students are very career-focused. We seek to maintain the program's technical quality and strengthen its ability to serve students in reaching their goals.

The department has also engaged broadly in recruitment, retention and student success initiatives. These include individual course redesigns, panels in CS 101, adoption of affordable learning solutions, faculty professional development, and new advising strategies. For example, the CS chair formerly handled all student and transfer advising; in the face of program growth, this became an unsustainable arrangement and has evolved into one where advising is distributed across the CS faculty. The faculty acknowledge that several courses stand out within SST for high DFW rates. These phenomena are not incongruous with nationally-observed DFW trends in early programming classes. That said, there are opportunities for improvement and troubling equity gaps. We should compare ourselves to our peer CSUs; however we have not been resourced appropriately to accept any ambitious goal like "achieve the smallest equity gaps among the CS programs across the CSU." In fact, we have among the smallest number of faculty, the highest student-to-advisor ratios, and ???? among our peer institutions across the CSU.

Building off our successful alumni reunion in 2013, the department has institutionalized a SSU CS Alumni LinkedIn group that now has over 400 current and past students. Our CS alumni have been a valuable source of adjunct faculty for the program, a resource for student careers, and a source for speakers in our colloquium series. The department sees value in developing more regular program assessment involving recent graduates and alumni. Existing efforts are intermittent and voluntary, as a form of irregular department service. The department should develop regular assessment strategies that do not depend on faculty volunteerism or worsen faculty workload. Such strategies could include leveraging staff support for senior exit surveys, warehousing assessment-related data, and housekeeping related to alumni contacts.

Our Senior focus group and alumni surveys have directly expressed the desire to see more substantial capstones, opportunities for experience as graders or TAs, and support for internship experience. The department should be responsive to this mandate. Historically low participation in internships via CS 497 suggests opportunities for growth and possible administrative barriers warranting investigation. Ideas for strengthening partnerships with industry and alumni include: investigate administrative barriers in organizing internships; investigate administrative barriers inhibiting collaboration with industry in capstones and other department activities; seek formal

partnerships with the SSU IT department in finding student positions that would credential those CS majors interested in IT careers; and find new opportunities for cooperation with alumni and partners in business and industry.

6.3 Plan an Overdue Curricular Redesign

The last major curricular redesign of the Bachelor's of Science in Computer Science program at Sonoma State was in 2007, started by CS Chair Dr. Ledin and completed by CS Chair Dr. Stauffer. The faculty of the department are in resounding agreement that a major revision to the existing curriculum is overdue. The ability to forgo major curricular redesign for nearly two decades, however, is a testament to the 2007 curriculum revision and the longevity of the core ACM CS Standards on which it was based.

Since the last program review, the department has largely innovated within the 2007 curriculum's framework. This predominantly involved the development of new CS electives. As part of this self-study, some curricular bottlenecks and areas of improvement have been identified, notably:

- CS 210 has been identified as a course that tends to impact the graduation timeline of our transfer population, who often do not arrive having satisfied an equivalent course, despite course articulations with their origin institutions. More generally, we may consider large changes in how we integrate transfer students into our curriculum. Their educational preparation appears to vary more than those of our FTF students when entering the program, commonly in terms of CS 215, CS 315, and CS 351 preparation.
- The effectiveness of CS 242 (or the adequacy of CS 242 to prepare students for later 400-level courses like CS 454 and CS 415) has been called into question. A full redesign of CS 242 is constrained by the department's goal of maintaining the articulation with California Community college courses under C-ID designation COMP 152, but innovation at the upper-division level may be possible. Likewise, the role of the "Math support courses" within the curriculum, which may not be adequately supporting student success in math-intensive CS courses.
- It may improve program assessment (and aid students during their job search) if internships, senior research projects and capstones were structured around the collection of one (of several) target artifacts. Designing these artifacts in a manner that supports university-level assessment of student writing would be convenient, due to the imminent necessity to satisfy the Graduation Writing Assessment Requirement (GWAR) outside the WEPT.
- The department is open to large-scale curriculum redesign of its core: rethinking the structure of its programming sequence (CS 115, CS 215, CS 315) and the relationship of this sequence with other courses in the core; re-design of CS 470 as a year-long capstone, similar to how it is offered at other CS programs in the CSU; the development of a CS-0 course or a "stretch CS 115" to improve FTF experiences in the major; the development of concentrations or tracks within the major (a "data science" track or a "CS

secondary teaching” concentration); and the development of upper division GE courses geared toward serving the university at large and/or a general STEM population.

Barriers to major curriculum changes include, of course, recent staffing disruptions and lack of time for any “extra” department service (especially in light of the significant school and university service already performed by its faculty).

6.4 Align Program Growth with Institutional Resources

Facing the steady growth of the CS major at SSU throughout 2009–2019, our department perennially discussed new opportunities to address the needs of students seeking study in computational fields outside the Computer Science major/minor. Those opportunities included a minor in data science (possibly in collaboration with the Math & Statistics department and others); a Bachelor of Arts in Computer Science degree; and/or the creation of bi-disciplinary or interdisciplinary CS+X tracks. Upper-division GE courses in CS could complement a variety of majors at Sonoma State. In particular, at various points in time, the department has discussed GE offerings to serve the campus including “Computer Ethics and AI,” “Beginning Data Science,” “Cyberwarfare & the Rise of the Hackers,” and “Human-Computer Interaction and Digital Design.”

Without resources to serve the university through GE offerings, using our existing resources to serve our majors takes precedence. For example, rather than respond to calls for new upper-division courses after the recent GE program revision, we retired CS 115 as a GE offering. Until we believe our growth can be supported by the institution, we need to stay within our resources—which does not allow innovation through the types of new interdisciplinary program offerings that may benefit Sonoma State’s recruitment of new students and growth. Ultimately, working within existing resources may require adaptations of the BS CS program to serve students within our capacity.

6.5 Address Challenges in Staffing and Scheduling

Over the period of review, the department's response to program growth has been to move from offering upper-division courses once a year to more than one section each semester. We also added a number of new courses, to draw on new faculty expertise and keep the curriculum current (CS 330, CS 425, CS 391, etc). Our alumni survey highlights some student disappointment about elective availability, which is the unfortunate consequence of introducing new electives and the realities of staffing. To facilitate increasing the number of courses/sections to keep up with demand, we both expanded our lab space (to a new computer lab in Stevenson) and increased our utilization of Darwin labs (by arranging more evening class times). Even with the addition of a new lab, we could not easily meet all CS demand for teaching spaces, tutoring spaces, club spaces, and hours designated for student-use in projects and capstones.

This self-study has revealed that it is the case that both tenured and tenure-track faculty in CS have routinely taught beyond their contracted teaching load in response to department needs.

The department has increased section sizes beyond previously agreed curricular limits in service to student graduation timelines. Faculty have stretched themselves to both teach early morning labs and late evening labs, due to scheduling necessity. Together, these historic practices may be a contributor to burn-out / attrition.

Staffing instability poses an imminent threat to the health and longevity of the CS program. Any retirement or faculty loss today would most likely necessitate one or more of the following program responses: accepting a larger percentage of non-CS courses as substitutes for major requirements (resulting in graduates exposed to fewer CS topics and the overall dilution of the technical rigor valued by our students); administratively rejecting requests to repeat courses more than twice (prematurely ending the careers of some students in the major); reducing the frequency of course offerings (increasing the timeline to graduation for our transfer population and any CS students repeating a course); and declaring impaction (artificially restricting the growth of the major, negatively impacting students who are historically underrepresented in CS). These options each need to be carefully considered before acting, with the goal of making it possible to offer our curriculum with the limited faculty that we currently have, to serve as many students as possible.

Bibliography

[ABET23] ABET. Criteria for accrediting computing programs, 2023 – 2024. Online: https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2023-2024/

[ACM21] ACM Data Science Task Force. Computing Competencies for Undergraduate Data Science Curricula. January 2021. https://doi.org/10.1145/345353

[Bennedsen19] Jens Bennedsen and Michael E. Caspersen. 2019. Failure rates in introductory programming: 12 years later. ACM Inroads 10, 2 (June 2019), 30–36. https://doi.org/10.1145/3324888

[Bennett20] Zackary Bennett. Why colleges are offering data science programs. *US News and World Report*, September 18, 2020. Online: https://www.usnews.com/education/best-colleges/articles/why-more-colleges-are-offering-data-science-programs

[Berkeley18] UC Berkeley Public Affairs Office. Berkeley offers its fastest-growing course – data science – online, for free. *Berkeley News*, March 29, 2018. Online: https://news.berkeley.edu/2018/03/29/berkeley-offers-its-fastest-growing-course-data-science-online-for-free/

[Brodley19] Carla Brodley, Randy Bryant, Lori Clarke, Juan Gilbert, Susanne Hambrusch, Chris Johnson, Richard LeBlanc,John Paxton, and Bobby Schnabel. Creating institutional homes for computing: Transforming a department into a school or college. Computing Research Association white paper, April 2019. Online: https://cra.org/resources/creating-institutional-homes-for-computing/

[Cai19] Carrie J. Cai and Philip J. Guo. Software developers learning machine learning: Motivations, hurdles, and desires. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2019, pp. 25-34. https://doi.org/10.1109/VLHCC.2019.8818751

[Chakraborty21] Partha Chakraborty, Rifat Shahriyar, Anindya Iqbal, and Gias Uddin. How do developers discuss and support new programming languages in technical Q&A site? An empirical study of Go, Swift, and Rust in Stack Overflow. In *Information and Software Technology*, Volume 137, 2021, 106603, ISSN 0950-5849, https://doi.org/10.1016/j.infsof.2021.106603

[D'Agostino22] Susan D'Agostino. Computer science's challenges, as seen by its pioneers. In *Inside Higher Ed*, September 30, 2022. https://www.insidehighered.com/news/2022/09/30/pioneers-discuss-challenges-facing-computer-science

[Guzdial19] Mark Guzdial. The growing tension between undergraduate and K-12: Is CS for all, or just those who get past the caps? *Communications of the Association for Computing Machinery (CACM)* blog, February 3, 2019. https://cacm.acm.org/blogs/blog-cacm/234489-the-growing-tension-between-undergraduate-and-k-12-is-cs-for-all-or-just-those-who-get-past-the-caps/fulltext

[Hey09] Tony Hey et al. eds: *The Fourth Paradigm: Data-Intensive Scientific Discovery.* Microsoft Research, Redmond, Washington (2009).

[Kafka20] Alexander C. Kafka. The discipline that is transforming higher ed. In *Chronicle of Higher Education*, April 15, 2020. https://www.chronicle.com/article/the-discipline-that-is-transforming-higher-ed/

[Ko06] Amy J. Ko, Brad A. Myers, Michael J. Coblenz and Htet Htet Aung. An exploratory study of how developers seek, relate, and collect relevant information during software maintenance tasks. In *IEEE Transactions on Software Engineering*, vol. 32, no. 12, pp. 971-987, Dec. 2006. https://10.1109/TSE.2006.116

[Lazowska14] Ed Lazowska, Eric Roberts, and Jim Kurose. “Tsunami or sea change? Responding to the explosion of student interest in computer science.” Presentation at NCWIT 10th Anniversary Summit, May 2014 and CRA Conference at Snowbird, July 2014. Online: http://lazowska.cs.washington.edu/NCWIT.pdf

[Levin23] Matt Levin. “Red Texas is flush, blue California has a deficit, but it’s more about economics than politics.” *Marketplace*, January 11, 2023. Online: https://www.marketplace.org/2023/01/11/why-california-has-budget-deficit-texas-has-surplus/

[Liu19] Jasmine Liu. “CS in Crisis: Is Stanford doing enough to respond to capacity and inclusion challenges?” *Stanford Daily*, February 9, 2019. Online: https://stanforddaily.com/2019/02/19/cs-in-crisis-is-stanford-doing-enough-to-respond-to-capacity-and-inclusion-challenges

[Lohr22] Steve Lohr and Tripp Mickle. As Silicon Valley retrenches, a tech talent shift accelerates. In *New York Times* (online), December 29. 2022. https://www.nytimes.com/2022/12/29/business/silicon-valley-tech-talent-mainstream-industries.html

[Manke18] Kara Manke. Berkeley inaugurates Division of Data Science and Information, connecting teaching and research from all corners of campus. In *UC Berkeley News*, November 1, 2018. Online: https://news.berkeley.edu/2018/11/01/berkeley-inaugurates-division-of-data-science-and-information-connecting-teaching-and-research-from-all-corners-of-campus/

[Murphy22] Austin Murphy and Martin Espinoza. Existential threat: Sonoma State’s budget crunch poses a bigger problem than the scandal surrounding its president. In *Santa*

Rosa Press Democrat (online), April 30, 2022. https://www.pressdemocrat.com/article/news/existential-threat-sonoma-states-budget-crunch-poses-a-bigger-problem-tha/

[NASEM18] National Academies of Sciences, Engineering, and Medicine. 2018. Assessing and Responding to the Growth of Computer Science Undergraduate Enrollments. Washington, DC: The National Academies Press. https://doi.org/10.17226/24926

[NASEM18-DS] National Academies of Sciences, Engineering, and Medicine. 2018. Envisioning the Data Science Discipline: The Undergraduate Perspective: Interim Report. Washington, DC: The National Academies Press. https://doi.org/10.17226/24886

[Newsome22] Melba Newsome. Computer science has a racism problem: These researchers want to fix it. In *Nature*, vol. 610, pp. 440-443, Oct. 2022. https://doi.org/10.1038/d41586-022-03251-0

[Patitsas16] Elizabeth Patitsas, Jesse Berlin, Michelle Craig, and Steve Easterbrook. Evidence That Computer Science Grades Are Not Bimodal. In *Proceedings of the 2016 ACM Conference on International Computing Education Research (ICER '16)*. https://doi.org/10.1145/2960310.2960312

[Pérez-Quiñones20] Manuel A. Pérez-Quiñones. What can CS departments do? Blog post, June 10, 2020; retrieved November 3, 2022. https://maperezquinones.medium.com/what-can-cs-departments-do-925aa4ade70f

[Pierson20] Emma Pierson, Elissa M. Redmiles, Leilani Battle, and Jessica Hullman. If you want more women in your workforce, here's how to recruit. In *Nature* Career Columns, August 26, 2020. https://doi.org/10.1038/d41586-020-02489-w

[Pirolli95] Peter Pirolli and Stuart Card. Information foraging in information access environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*. ACM Press/Addison-Wesley Publishing Co., USA, 51–58. https://doi.org/10.1145/223904.223911

[Roberts11] Eric S. Roberts. Meeting the challenges of rising enrollments. *ACM Inroads* 2, 3 (September 2011), 4–6. https://doi.org/10.1145/2003616.2003617

[Roberts16] Eric Roberts. A history of capacity challenges in computer science. March 7, 2016. Online: https://cs.stanford.edu/people/eroberts/CSCapacity/

[Shrestha20] Nischal Shrestha, Colton Botta, Titus Barik and Chris Parnin, Here we go again: Why is it difficult for developers to learn another programming language? In *IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, 2020, pp. 691-701.

[Schvartzman] Lara Schvartzman. Representation and what happened to women in tech. Online article from LenioLabs, retrieved November 3, 2022. https://research.leniolabs.com/women-in-tech/

[Singer19] Natasha Singer. The hard part of computer science? Getting into class. In *New York Times* (online), January 24, 2019. https://www.nytimes.com/2019/01/24/technology/computer-science-courses-college.html

[Tamer18] Burçin Tamer. Are Ph.D. students losing Interest in faculty positions at research universities? In *Computing Research News*, vol. 30, no. 1, January 2018. https://cra.org/crn/2018/01/ph-d-students-losing-interest-faculty-positions-research-universities/

[WASC21] WASC Senior College and University Commission. 2013 Handbook of Accreditation, revised November 2021, Section 4: Educational Quality: Student learning, core competencies, and standards of performance at graduation. Online: https://www.wscuc.org/handbook/#4--educational-quality--student-learning--core-competencies--and-standards-of-performance-at-graduation

[WillsCRA22] Craig E. Wills. Outcomes of advertised computer science faculty searches for 2022. In *Computing Research News*, vol. 34, no. 10, November 2022. https://cra.org/crn/2022/11/outcomes-of-advertised-computer-science-faculty-searches-for-2022/

[WillsTR22] Craig E. Wills. Outcomes of advertised computer science faculty searches for 2022. Worcester Polytechnic Institute Technical Report WPI-CS-TR-22-05, October 2022. https://web.cs.wpi.edu/~cew/papers/outcomes22.pdf

[Wingfield17] Nick Wingfield. The disappearing American grad student. In *New York Times* (online), November 3. 2017. https://www.nytimes.com/2017/11/03/education/edlife/american-graduate-student-stem.html

[Zweben21] Stuart H. Zweben, Jodi L. Tims, Cindy Tucker, and Yan Timanovsky. ACM-NDC Study 2020-2021: Ninth annual study of non-doctoral-granting departments in computing. In *ACM Inroads*, vol. 12, no. 4, December 2021. https://www.acm.org/binaries/content/assets/education/acm_ndc_2020-2021.pdf

[Zweben22] Stuart Zweben and Betsy Bizot. 2021 Taulbee Survey: CS enrollment grows at all degree levels, with increased gender diversity. Computing Research Association report, May 2022. https://cra.org/wp-content/uploads/2022/05/2021-Taulbee-Survey.pdf

Appendices

Appendix A — Overview

- CS Program Review 2009
- SSU Program Review Policy

Appendix B — Assessment

- CS Alumni Survey Instrument
- Report on CS Senior Focus Group session, Spring 2022

Appendix C — Faculty CVs

- Glenn Carter — lecturer
- Ying Lin, PhD — lecturer
- Johnathan Thompson — lecturer
- Henry Walker, PhD — lecturer
- Gurman Gill
- Mark Gondree
- Ali Kooshesh
- B. Ravikumar
- Suzanne Rivoire

Appendix D — Catalog and Course Offerings

- SSU Catalog, 2022–2023, BS in Computer Science
- SSU Catalog, 2022–2023, minor in Computer Science
- SSU Catalog, 2022–2023, CS Course Descriptions
- CS Course List, Fall 2022
- CS Course List, Spring 2023

Appendix E — Curriculum

- Prerequisite Chart for BS in CS, effective Fall 2022
- Comparison of SSU CS PLOs with others
- Comparison of unit Requirements in B.S. CS Programs

Appendix F — Department Programs & Partner Programs

- CAHSI Memorandum of Understanding
- MESA, Updates from SST School Meeting for 2022-2023
- TIPS Toward Justice Grant program (2020-2025) information from ORSP
- SST Women in Tech Group (2016-2022), website
- SST Sonoma Mountain Connection Program (2021-2023), website
- Sonoma Mountain Connection Spring 2022 e-portfolios website
- CS Club event, NomaHacks (2019), website
- CS Colloquium flyers

Appendix G — Facilities and Program Resources

- Technology Classrooms & Conference Rooms

Appendix H — Course Syllabi

- CS 101, Computing Technology and You, Fall 2022
- CS 115, Programming I, Fall 2022
- CS 210, Introduction to Unix, Fall 2022
- CS 215, Programming II, Fall 2022
- CS 242, Discrete Structures for Computer Science
- CS 252, Intro to Computer Organization, Spring 2022
- CS 315, Data Structures, Fall 2022
- CS 340, Computer Security and Malware, Spring 2022
- CS 351, Computer Architecture, Fall 2022
- CS 355, Database Management Systems Design, Fall 2022
- CS 370, Software Design and Development, Fall 2022
- CS 390, Computer Science Colloquium, Fall 2022
- CS 391, Computing Professions, Fall 2022
- CS 415, Algorithm Analysis, Fall 2022
- CS 450, Operating Systems, Fall 2022
- CS 454, Theory of Computation, Fall 2022
- CS 460, Programming Languages, Fall 2022
- CS 470, Advanced So!ware Design Project, Fall 2022
- CS 479, Fundamentals of Computer Vision, Fall 2022
- CS 480, Artificial Intelligence, Fall 2022
- CS 495, example Special Studies contract courses
- CS 496, example Senior Research Project contract courses