Case for Promotion to Full Professor

In real estate there is a principle called "Highest and Best Use" that concisely describes how land should be utilized. At a high level, this principle argues that warm, sandy oceanfront property should be used for housing or recreation rather than strip malls.

During my time at GVSU (and especially since receiving promotion and tenure in 2010), I have made a point of putting my time and talents to their "Highest and Best Use." Specifically, I have made a point of choosing how to spend my limited time based on what will be most beneficial for our students.

Summary

Below is a brief outline of how I have met the criteria for promotion to full professor given in the 2025 College of Computing Personnel Document:

Teaching (Excellent):

- *Strong demonstrated interest and proven ability in high-quality and best practices in computing instruction and curriculum/laboratory development*
  - I led the push for increased emphasis on teaching testing throughout the curriculum – especially in CS 1 and CS 2. Many of my publications over the years have been related to this emphasis.
  - I was one of the first computing instructors to adopt alternative grading practices (e.g., "mastery grading" and other principles from *Grading for Growth*), and the first to promote it strongly to peers.
  - Both of my sabbaticals were focused on gaining "real world" industry experience to improve my effectiveness as a teacher and mentor.
- *Consistently received positive student evaluations/feedback and peer evaluations*
  - My quantitative student evals are consistently above 4.0 on a 5.0 scale.
  - The reports from my formal classroom visits are overall positive and do not indicate any major concerns. Any concerns raised are addressed in detail in my annual activity reports.
  - I was nominated for the GVSU Outstanding Teacher Award in 2022.
- *Very active in significant revision/improvement of course materials or the development of new programs or courses or labs*
  - As CIS 163 coordinator, I redesigned course projects to include testing as a fundamental component.
  - I led the re-design of CIS 251 and CIS 451 from two required courses to a required course and an elective course (to increase flexibility in the curriculum)
  - I re-designed CIS 351 and CIS 371 to use mastery grading and have a significant "flipped" component.
  - I re-designed CIS 500 (both content, delivery, and evaluation) in response to (a) the number of incoming students who had little or no programming experience, and (b) the number of students who were managing to complete assignments without meeting the learning objectives (i.e., students would submit working code but still could not program effectively.)
- *Delivery of high-quality content in multiple courses at different levels*

- During my time at GVSU, I have taught a variety of courses ranging from introductory programming (CIS 162 and 163) to grad courses (CIS 500 and CIS 658). Other courses include CIS 251/351, CIS 451, CIS 263, CIS 343, and CIS 371.
- I developed a complete set of high-quality videos for CIS 351.
- My course evals, peer-evals, and resulting reflections in my annual activity reports demonstrate the high-quality of my course content.
- My course materials from 2018 onward are online: https://kurmasgvsu.github.io/

Scholarship (Accomplished):

- *Demonstrated establishment of sustained research plan*
  - The majority of my scholarly activity has focused on developing tools to help students and instructors be more efficient. My goal is to (a) help students learn material more quickly and/or with less cognitive load and (b) to help instructors complete the "mechanical" aspects of teaching quicker and easier to have more time and energy to devote to student interactions. Almost all of my activity in this area has resulted in a peer-reviewed publication.
- *Peer-reviewed publications such as papers in journals, conferences, workshops, and books*
  - I have averaged at least one peer-reviewed publication per year during my entire time at GVSU. (A complete list is included later in this document.)
- *Supervision of graduate and/or undergraduate research students*
  - I have supervised both independent undergraduate projects, as well as capstone and master's projects. (Examples included later in this document.)
- *Secured research funding*
  - 2008: FTLC Summer Research Grant
  - 2018: Secured donation through my sabbatical to fund future undergrad projects
  - 2024: SIGCSE Special Projects grant

Service (Accomplished):

- *Significant contributing to the academic/professional community through responsibilities such as journal editorships, conference program/organization committee memberships, grant proposal review panels, K-12 outreach, panels in the community related to education/tech, and other professional society organizers*
  - WCAE Program Committee 2018, 2023, and 2025.
  - Site Host for CCSC:MW 2024
  - Associate Chair for CCSC:MW 2025
  - Event Supervisor for CodeBusters at Science Olympiad State Finals 2024.
- *Leadership roles and active services on taskforces/committees and participation and engagement in Department, College and/or University wide committees, volunteering for local STEM events and K-12 outreach, meetings and events*
  - Member of Faculty Salary and Budget Committee since 2006 (including significant subcommittee work on improving the salary increment process and working with Provost Davis to improve the sabbatical application/approval process).
  - Led the revision of the scholarship application process for those scholarships awarded by the Student Affairs Committee.
  - Member of the COACHE steering committee. (This may be my most significant service contribution while at GVSU. See my 2024 Activity Report for details.)
  - Member of several committees related to our transition to a College.

Teaching Details / Evidence

Teaching Statement (T1)

Teaching is, without question, our most important priority. Our students are best served by taking well-taught, rigorous courses that hold them to high standards. Instructors who maintain high standards must provide a correspondingly high level of service. I make a point of being highly available to students to better support them as they work through challenging assignments. For example, I make a point of being available in my office whenever I'm not in a meeting or teaching. Furthermore, I make a point of choosing scholarship, service, and sabbatical projects that directly benefit students.

I use proficiency grading in all my courses. My main motivation for using this approach is to encourage students to become proficient in as many course topics as possible, even if they require significant improvement in a few course topics. In contrast, traditional grading (i.e., a "pile of points") encourages students to do just enough with each course topic to accumulate the desired number of points. As a result, it is not unusual for students who earn a "C" or "D" in a traditional course to leave that course without any usable skills (they can get everything partially correct), whereas a "C" or "D" student from an alternatively graded course is more likely to have acquired several useful skills. (For more details, see https://www.youtube.com/watch?v=65U6CJ2gvLY, *Grading for Growth* by Talbert and Clark, and *Specifications Grading* by Linda Nilson.)

Over the years, two main concerns stand out in my student and peer evaluations: (1) the difficulty of my courses (or the time needed to be successful), and (2) the struggles some students had with proficiency grading. To keep the workload on my courses reasonable, I am constantly evaluating the "efficiency" of my assignments to make sure the pedagogical benefits of completing the assignment are sufficient given the time needed to complete the assignment. The growth of the College has helped with this task because most courses are now taught by a wider variety of instructors. As a result, I am better able to adjust my assignments to be more in line with others teaching the same course.

Concerns about my use of proficiency grading have decreased significantly over the past two years. This decrease is a result of two factors: (1) Most of my teaching has been in CIS 500, and proficiency grading is especially well suited to "skill-based" courses like 500; and (2) I have moved from grading quizzes pass/fail to awarding points/partial credit.[1] This change has significantly reduced the stress some students feel because most struggling students now see a projected passing grade on their progress report rather than the "F" they were seeing previously. This change allows them to more clearly see that they are on track to pass the course, which makes them less likely to needlessly drop.

Overall, my student evaluations have been much stronger during the past two years. (Keep in mind that graduate students tend to give more positive evaluations than undergraduates, so a direct comparison is not possible.)

The following sections (especially "Student Evaluations") contain more evidence of my effectiveness as an instructor.

[1] In courses other than CIS 500.

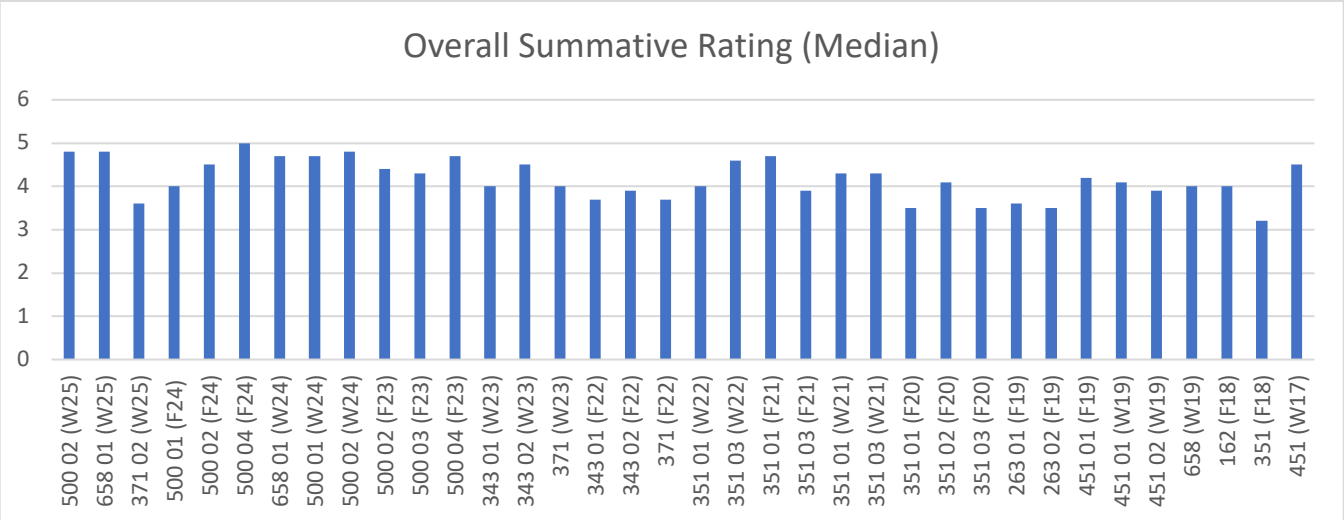
Courses Taught 2018-2025 (T3)

- CIS 162 – Computer Science I (4 credits)
  - F18
- CIS 163 – Computer Science II (4 credits)
  - S20, S21
- CIS 263 – Data Structures (3 credits)
  - F19 (x2)
- CIS 343 – Programming Languages (3 credits)
  - F22 (x2), W23 (x2)
- CIS 351 – Computer Organization (4 credits)
  - F18, F20 (x3), W21 (x2), F21 (x2), W22(x2)
- CIS 371 – Web Programming (3 credits)
  - F22, W23, W25
- CIS 451 – Computer Architecture (3 credits)
  - W19 (x2), F19
- CIS 500 – (3 credits)
  - F23 (x3), W23 (x2), F24 (x3), W25
- CIS 658 – Web Architectures (3 credits)
  - W19, W20, W24, W25
  -

Student Evaluations (T4)

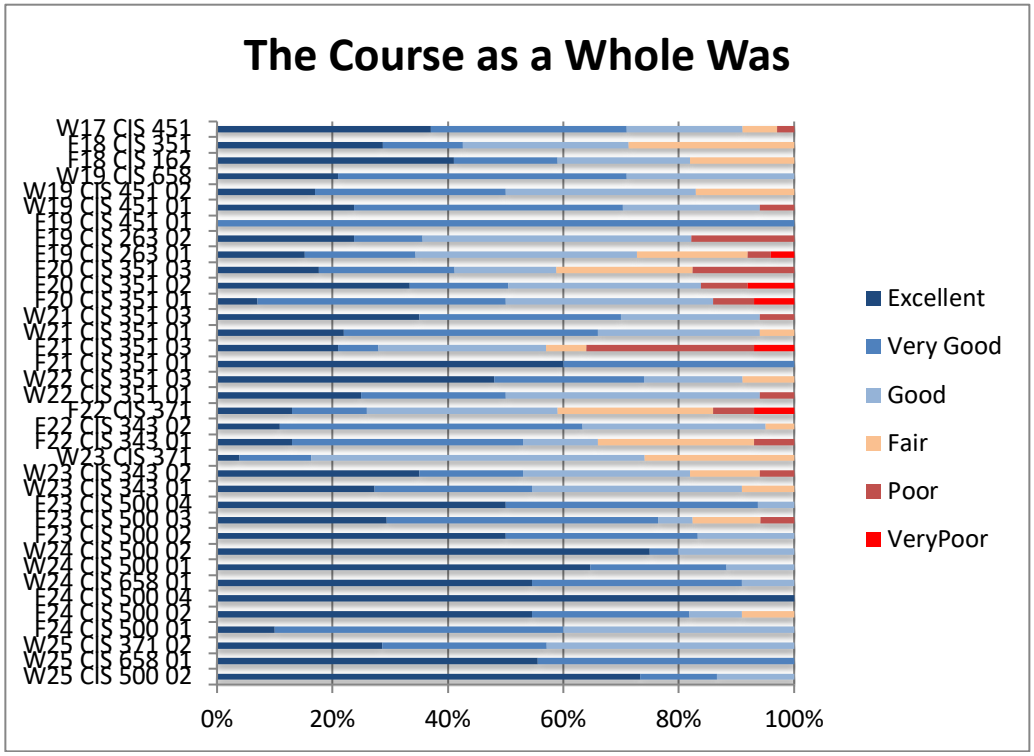
I make a point of maintaining high expectations for students but also providing the necessary support to help students meet those expectations. When viewed as a whole, both student and peer evaluations show that I am an effective instructor.

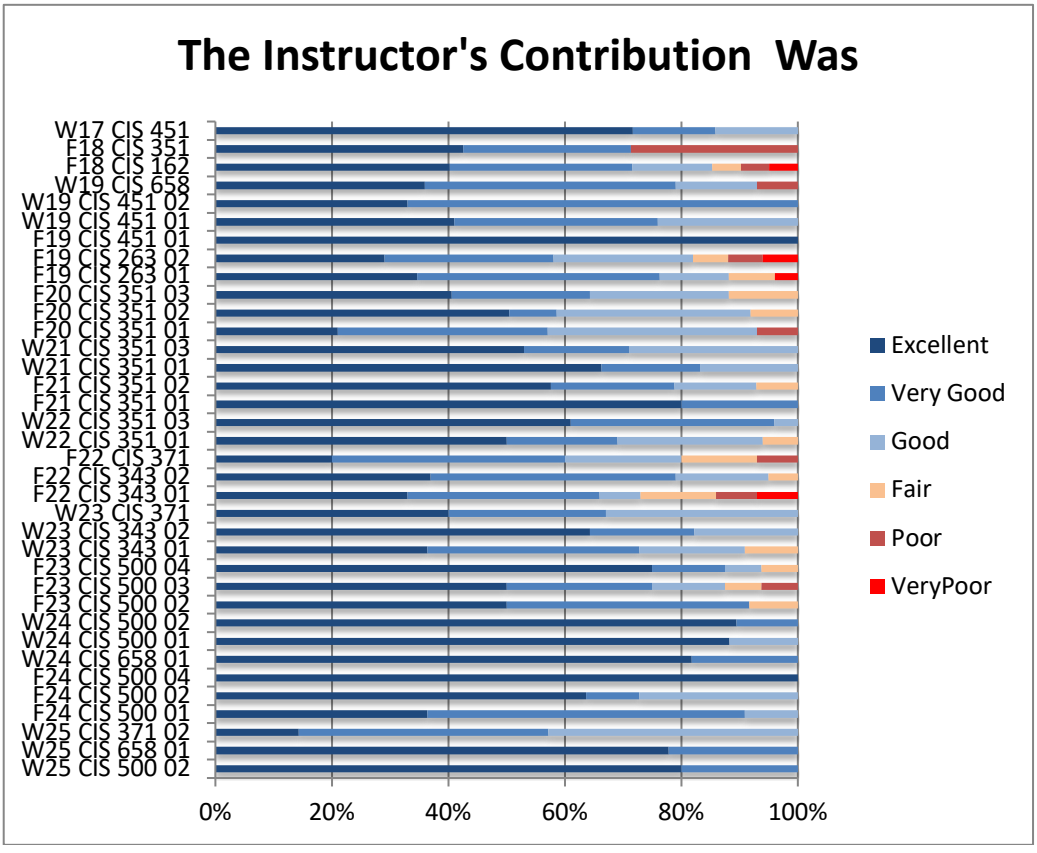
The "overall summative rating" of my LIFT evaluations are often at or above 4.0 and, as can be seen below, have fallen below 3.5 only once since 2017. Lower scores tend to be correlated with new preps and/or big changes like my switch to proficiency grading.[2]



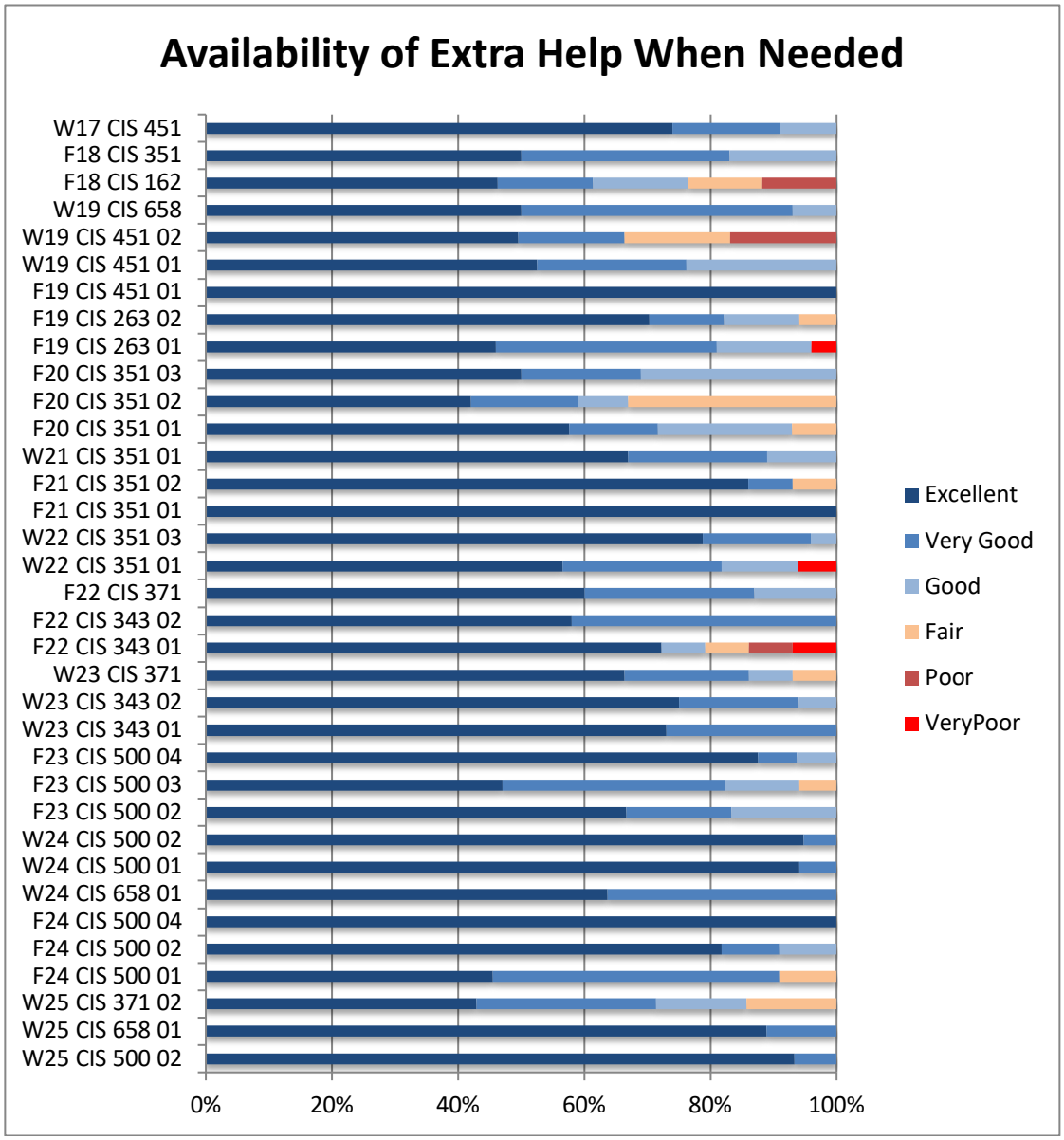
[2] Thanks to Jonathan Leidig for sharing the tools to prepare these charts.

A strong majority of students choose “Excellent”, “Very Good”, or “Good” when asked “This course was:” and “The instructor’s contribution to this course was:” Notice that there are noticeably more “blue” marks in the chart focusing specifically on my contribution to the course.





Because I have high expectations, I make a point of being highly available to my students. In 27 out of 35 sections since 2017, at least 80% of students responded with “Excellent” or “Very Good” when asked whether “Extra help was readily available.”



Free response comments tend to be positive. The most common concerns raised are

1. Overall workload, and
2. Struggles with proficiency grading.

In response:

1. I re-evaluate every assignment every semester to make sure students get a reasonable return on the time they invest.

2. When using proficiency grading, I have returned to awarding point (and partial credit) on quiz questions so that quizzes don't have the "all or nothing" feel that appears to have caused students the most stress. When quizzes are only marked pass/retry, some students see an overall failing grade on their progress report, even though they will likely earn a passing grade by the end of the semester. Reverting to points and partial credit makes it easier for these students to see they are likely to pass the course, reducing their stress and reducing the likelihood they will needlessly withdraw from the course. The partial credit also makes quiz retakes feel more like an "optimization" than a necessity, which also reduces stress.

(See reflections in annual activity reports for details)

Curriculum Development (T5)

I led the reorganization of CIS 251 and CIS 451 from two required courses to one required course and one elective. As part of this reorganization, the three-credit CIS 251 became a four-credit CIS 351 with a lab. A significant part of the reorganization was determining which topics from the four-credit CIS 451 would be moved into the new CIS 351.

I have also been working with Erin Carrier and others to redesign the math cognates for the Computer Science major.

Capstone / Senior Project / Supervised Projects (T6)

- Brian Mbeere (Master's Project 2019)
  - Brian wrote an app that used GPS sensors and QR codes to help Steelcase track and manage trailers while on Steelcase property making deliveries.
- Colin Beaudoin (Independent Study 2019)
  - I supervised Colin's independent study into quantum computing. Colin is now completing his Ph.D. in quantum computing at Penn state.
- Vincenzo Pavano (Master's Project 2020)
  - Vincenzo wrote an app to help foreign language students build vocabulary.
- Justin Wickenheiser (Master's Project 2020)
  - Justin wrote a web app to help instructors write marching band drill.
- Spencer Firlik (Master's Project 2020)
  - Spencer wrote a web app to dynamically generate practice problems for the combinatorial circuits module of a computer organization class.
- Samuel Quist, Selena Cade, Ryan S., Nicole Kinney (IS capstone 2023)
  - This team wrote an application to collect faculty teaching preferences to help automate the creation of course and teaching schedules.
- Anna Carvalho (Supervised Project 2023)
  - Anna helped me with the Poprawa project --- a gradebook for alternatively graded courses (https://dl.acm.org/doi/10.1145/3626253.3635488)
- Hari Kalahasti (Master's Project 2024)
  - Hari wrote a web app to help recent graduates track their job applications.
- Teresia Nduati (Master's Project 2025)
  - Teresia wrote a web application to help asthma patients track their health and correlate their well-being with external factors such as the weather.
- Jordan Bancino, Austin Hargis, Aaron MacDougall (CS Capstone Project 2025)

    - This team wrote a simulator-independent unit test framework for simulated digital logic circuits.
- Hannah Woznick, Juan Hernandez, Anthony Lazo, Darryl Gee (CS Capstone Project 2025)
    - This team wrote an iPad app to allow instructors to easily grade student submissions when they are submitted using GitHub repositories.
- Ian McCourt, Catherine Stacey, Gavin Halstead (CS Capstone Project 2025)
    - This team wrote a web application that will track foreign language reading selections, and also use them to (a) provide translation hints when necessary, and (b) build flashcards.
- Hilda Ogamba and Benedict Ofesi (Supervised Project 2025)
    - Hilda and Benedict helped me prepare `gvQLC`, a VS Code extension to help instructors efficiently prepare custom questions based on submitted code.

Course Redevelopment (T8)

CIS 500: During Fall 2022 and Winter 2023, I (with input from several colleagues, including Robert, Guenter, Sammah, and Suhila) made significant changes to CIS 500. Prior to F23, the course was a combination of CS 1 and CS 2. However, that approach was not serving students in our Cybersecurity, Data Science and HBIN programs well. In addition, there was *a lot* of cheating. In response,

- I de-emphasized "ACS-focused" topics like recursion and data structures and replaced them with topics beneficial to students in all programs including (a) Pandas, (b) plotting, and (c) regular expressions.
- I prepared two Jupyter notebook-based PrairieLearn assignments for each module: One that we work through together in class, and one for homework
- The primary component of the student's final grade is now the performance on weekly pencil-and-paper quizzes. (The return to pencil-and-paper quizzes significantly reduced the effects of cheating.)

CIS 351: I created a series of carefully scripted, polished videos for CIS 351 (https://tinyurl.com/cis351videos). I was by no means the first instructor in the School to create video content; but, to the best of my knowledge, I was one of the first to design the content specifically for video (as opposed to more-or-less recording the same lecture I would give in the classroom). Students frequently complement me on how helpful these videos are and how they appreciate that they are more than just a camera pointed at a lecture.

These videos cover about 80% of the content of CIS 351. I also have prepared videos for select topics in CIS 371 and CIS 343 (https://www.youtube.com/@prof.kurmas9240/Jordas).

Nominated for University Outstanding Teacher Award (T9)

In 2022, I was nominated for the Outstanding Teacher Award. I have included the following letters of support from the nomination packet.

- Ira Woodring: CIS faculty
- Scott Grissom: CIS faculty
- Cymbree Spoher: Current student (at the time)
- Christopher Carr: Former student

Other Significant Teaching Activities (T12)

Industry Placement Sabbaticals

For both of my sabbaticals, I spent a year working in industry.

Note: The sabbaticals themselves are technically professional development, not teaching. I discuss them here because they provide important context.

I spent the 2010-2011 academic year working at Atomic Object --- a consultancy with (at the time) about 30 developers. During this time, I learned what it was like to "craft" software professionally. For example, I learned about design techniques like dependency injection, and how "pair programming" worked in practice (as opposed to during CIS 162 labs). In my opinion, the most important thing I learned during this year is how central testing is to the entire software development process. Testing is not simply the last step before releasing code; but, when included from the beginning, is a process that helps guide the design and development of the software: Writing tests early both highlights potential oversights in the design process and helps identify difficulties that may arise during implementation.

I spent the 2017-2018 academic year doing web development for Spectrum Health. This experience was a nearly ideal complement to my experience at Atomic Object. For the most part, AO built applications from "the ground up." In addition, they were also fortunate enough to be selective about the projects they accepted. As a result, they were able to leverage the newest technologies as well as incorporate the current best practices. In contrast, my team's work at Spectrum Health was almost entirely to add features and/or fix large, legacy applications like MyHealth. Consequently, although they also looked for opportunities to improve their tools and processes, it wasn't always feasible to use best practices like fully automated tests. Spectrum Health taught me what it was like to develop code with many real-world constraints (as compared to AO's comparatively ideal setting). Among the many important things I learned during this sabbatical, two in particular stand out:

(1) I learned how the testing and deployment process can be carefully managed even when testing cannot be fully automated. This experience helped me improve the automation of my feedback to students to help them more efficiently correct mistakes and resubmit when appropriate.

(2) I learned what happens to our "C" students after graduation. While at Spectrum Health, I worked with 11 of my former students – most of whom were mediocre at best. In fact, the person who trained/mentored me failed CIS 451. However, all of these students were succeeding; most were thriving. These struggling students tended to fall into two groups (a) students who have a very specific interest, excel at that interest, but struggle in unrelated courses (especially CIS 351/451), and (b) students who simply weren't at a place in life where they were ready to focus on school.

My sabbatical experiences led to co-organizing a SIGCSE panel with Ryan McFall and a formal peer-reviewed publication with Mark Lewis and Lisa Lacher:

- SIGCSE Panel: R McFall, Z. Kurmas, P. Conrad, D. Frailey. Why and How to Spend a Sabbatical in Industry. SIGCSE '19: Proceedings of the 50th ACM Technical Symposium on Computer Science Education. February 2019
- M. Lewis, L. Lacher, Z. Kurmas. Industry Sabbaticals in Computing: Motivation and Preparation Computer Science and Computer Engineering (CSCE) 2024.

"Thought Leader / Early Adopter"

I have been the thought leader / promoter / early adopter of several important changes in the School of Computing:

Integrating testing into the curriculum

My time at Atomic Object taught me that testing is more than a check for bugs before deploying a product. Instead, testing should be an integral part of the software development process that is used to both direct the software design and help avoid bugs from being introduced in the first place. Upon returning to the classroom in 2011, I made a point of (1) having automated tests for as many assignments as practical (even "non-coding" assignments like building simulated digital logic circuits). I also began regularly encouraging students to follow a "Test Driven Development" workflow, where they write tests before writing code.

Today many instructors in the School of Computing use some form of automated testing for at least one assignment.[3] (See Hans Dulimartia's promotion packet.)

Relevant peer-reviewed papers:

- Z. Kurmas. Improving student performance using automated testing of simulated digital logic circuits. *Proceedings of the 13th annual conference on Innovation and technology in computer science education (ITiCSE 2008),* July 2008.
- Z. Kurmas. MIPSUnit: A Unit Testing Framework for MIPS Assembly. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education,* March, 2017.
- Z. Kurmas. DLUnit: A Unit Testing Framework for Simulated Digital Logic Circuits. *Proceedings of the 2023 IEEE/ASEE Frontiers in Education Conference (FIE),* October, 2023.

Using "DevOps" Tools to Automate Feedback

When I began using automated tests in my classroom, they had to be run/launched manually. Either I would download all student submissions, run their code against my tests, then email the results;[4] or, I would provide all the tests to the students so they could run the tests themselves.

While working at Spectrum Health, I saw how the DevOps team could set up a CI/CD pipeline that would automatically run automated tests whenever code was pushed to a git repository. Upon returning to teaching, I sought out ways to use similar industry tools in my classroom to provide feedback to students sooner. My first iteration was to use a web application called TravisCI; but soon after, I learned about (1) GitHub Classroom which automates the process of creating and configuring git repositories for students, and (2) GitHub Actions, which can be configured to run automated tests and report errors.

This automation has become especially beneficial now that I allow students to resubmit almost all assignments: It is much easier for students to fix bugs when they get feedback within a few minutes

[3] To be clear: I'm not claiming that everyone who uses automated testing at GVSU does so because of me. I'm primarily pointing out that I was one of the first adaptors and promotors in the School. Some colleagues adopted the practice based on conversations with me, others learned about similar processes from colleagues at other universities.

[4] At the time, being able to download student code from Blackboard then run it was only possible thanks to a very clever script written by Hans Dulimarta. Switching to GitHub and PrairieLearn has simplified that process significantly.

(while they are still thinking about the assignment) than when it takes a day or more to get feedback (and they have mentally moved on to other assignments/topics).

To the best of my knowledge, I was the first instructor in the School of Computing to make significant use of these tools. Others have since adopted similar processes. Some have followed my lead by adopting "professional" tools like GitHub; others have adopted similar practices using tools like PrairieLearn and zyBooks that are designed specifically for education.

I led a tutorial on using GitHub Classroom and GitHub Actions at CCSC:Midwest in 2021:

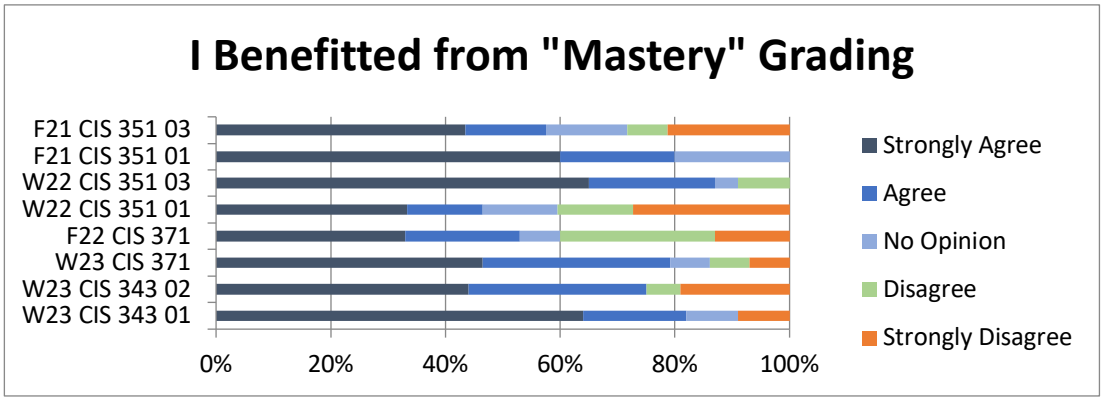
- Z. Kurmas. Using GitHub classroom for assignment management and automated feedback. Journal of Computing Sciences in Colleges, Volume 37, Issue 4, October 2021. (Tutorial)

Adopting principles from "Grading for Growth"

I've never been comfortable with the traditional "pile of points" system of grading. I could never get the "weight" right: I never figured out how to assign a percentage of points to things like homework and labs that was high enough to motivate students to complete them, but not so high that performance on these formative assignments dominated the summative assessments like tests. I also didn't like that once a student submitted an assignment there wasn't a strong motivation to fix misunderstandings and mistakes.

In 2020, I began reading blog posts by Robert Talbert and David Clark promoting alternate techniques for assessing students and assigning grades. Their ideas are largely extensions of those promoted by Linda Nielson, Joe Feldman, and others. Their arguments against traditional grading accurately described the problems I noticed in my classes. In Fall 2021, I introduced a grading system similar to that used by Talbert. (I called it "mastery-based" grading; but, that is not a very good descriptor for several reasons. I now use the term "proficiency grading".) I made mistakes the first few semesters; but, overall, a "Grading for Growth" approach is clearly more effective than traditional grading for *most* students. In addition, the changes I've made since F23 (see Section T4 above) have reduced the number of concerns significantly.

From F21 through W23, I included the following question on my LIFT evals: "I benefitted from the 'Mastery Grading' approach to this course" from F21 through W23. The chart below shows that a clear majority of students believed they benefited from this approach. In addition, work submitted on tests and quizzes as well as conversations with individuals strongly suggest that most students are learning better.



The downside is that, prior to F23, about 20% of students didn't believe they benefited from my implementation of proficiency grading. The main problem for many of these students appears to be the "snowball effect": A student who fails to meet an assignment's expectations must then take time to both address that assignment's deficiencies and keep up with new material. As discussed above, re-evaluating some assignments, and re-introducing partial credit to quizzes appears to have significantly reduced the number of students with concerns about proficiency grading.

It is too soon to tell what effect my efforts in "Grading for Growth" will have on the College of Computing; but, given how successful my approach has been in my classroom for the majority of students (especially after the changes I've made since F23), and given the growing popularity of "Grading for Growth" in other departments, and at other universities, I will be very surprised if many of our courses don't adopt at least a few of these ideas.

Relevant peer-reviewed paper:

- E. Spertus, Z. Kurmas. Mastery-Based Learning in Undergraduate Computer Architecture, Proceedings of the 21st Workshop on Computer Architecture Education, June 2021.

Videos

I should mention the series of videos I created for CIS 351 (https://tinyurl.com/cis351videos). I was by no means the first instructor in the School to create video content; but, to the best of my knowledge, I was one of the first to design the content specifically for video (as opposed to more-or-less recording the same lecture I would give in the classroom). Students frequently complement me on how helpful these videos are and how they appreciate that they are more than just a camera pointed at a lecture.

These videos cover about 80% of the content of CIS 351. I also have prepared videos for select topics in CIS 371 and CIS 343 (https://www.youtube.com/@prof.kurmas9240/Jordas).

Scholarship

Research Statement (R1)

I make a point of investing my scholarship time in projects that directly benefit our students. My goal is to use my scholarship as a tool to support my teaching and service. The main theme of my scholarly work at GVSU has been to develop tools and workflows to provide immediate, automated feedback to students. I have mainly focused on tools for use in CIS 251/351/451 because (1) these courses represent the majority of my teaching, and (2) there was less existing work / more room for improvement in this space. My most recent projects (`plnq` and `gvQLC`) have focused on CIS 500.

In my opinion, my most beneficial scholarly projects have been:

- `JLSCircuitTester` / `DLUnit` (unit testing frameworks for simulated digital logic circuits)
- `MIPSUnit` (a unit testing framework for MIPS assembly)
- `ICOS` (a "bare metal" framework to be used in Computer Architecture)

`MIPSUnit` was accepted as a full paper at SIGCSE. (SIGCSE is the most prestigious conference to which this work applies.) `ICOS` was published at the Workshop on Computer Architecture Education (which is the most relevant audience for this work). `DLUnit` has been accepted at Frontiers in Education (which is informally one step below SIGCSE/ITiCSE) which will be held in October.

I have included letters from two instructors describing how they have benefited from `MIPSUnit`.

I have also worked on "pure" research projects. Most notably, I worked with Jerry Scripps, Hugh McGuire, and Christian Trefftz on techniques for enumerating the different possibilities for assigning people to communities. This work led to several joint publications. My experience with this group demonstrates that I can contribute to "traditional" research; but, I find that I am more productive and fulfilled working on software-based, classroom-focused projects.

Finally, I have participated in two teaching-focused NSF projects (doing the work, not writing the grant)

- "Pencil Puzzles" led by Zach Butler and Iwona Bezakowa at RIT
- "Dive into Systems" led by Suzanne Matthews at West Point

Going forward, I have begun working on two main projects:

Questions over Learners' Code (QLC)

One challenge when teaching courses where programming projects are a significant component is determining whether a student truly understands the code he or she submitted. This problem has increased with the advent of tools like ChatGPT. One commonly suggested solution is to quiz students over the code they have submitted. However, this approach does not scale well: Historically, it has been too time consuming to prepare and administer a unique quiz for each student.

My students (Hilda Ogamba and Benedict Ofesi) and I have written a VS Code extension that allows instructors to write quiz questions customized to each student as the instructor reviews the submitted code. Assuming (1) section sizes of at most 40, and (2) that the instructor is already in the practice of reading through, and commenting on, all code submissions, we believe that writing quiz questions while writing comments will not add much time to the overall process. We are investigating techniques to

minimize the additional time required, such as (a) caching previously written questions (since the same question likely applies to several students) and (b) using generative AI to suggest questions.

Evaluating "Understand"-based learning objectives

Dave Musicant and I are looking into better/easier ways of assessing learning objectives that are focused on "Explain" and/or "Understand" --- especially in courses that allow reassessment without penalty.

The challenge, of course, is that simply asking student questions like "Explain how a ripple-carry adder works" either

- Rely on some element of surprise (which can be unfair to students), or
- Allow students to simply memorize the answer.

(We are especially concerned about the potential for memorization when students are offered reattempts without penalty.)

The common alternative is to morph the question into an "apply" question such as "The ripple carry adder in the diagram below has a broken wire (highlighted). Describe the set of inputs that will still generate correct outputs. Explain your reasoning." The challenge is that

- Depending on the topic, It can be difficult to think up good questions like this, and
- These type of questions will blindside some students --- unless they can practice them first (which can lead to the problem of students memorizing an algorithm rather than demonstrating a true understanding of the topic in question).

We are in the very early stages of this project and are working through the book *Taxonomy for Learning, Teaching, and Assessing, A: A Revision of Bloom's Taxonomy of Educational Objectives* by Anderson and Krathwohl and refining our research questions.

Awarded Funding (R2)

- FTLC Summer Research Grant (2008)
  - Small summer salary stipend to work on `JLSCircuitTester`.
- Student Support provided by Custom Business Solutions (2018-2019)
  - After completing my 2017-2018 sabbatical with Spectrum Health, I continued on as a contractor through Custom Business Solutions. During my time working for them, they donated about $8,000 for me to use to hire students to help with my scholarship.
- SIGCSE Special Projects Grant (2024)
  - I received a $3600 grant from SIGCSE to hire students to help me develop the `gvQLC` VS Code plug-in.

Publications Organized by Topic (R4)

JLSCircuitTester

For context, my first project in this area was `JLSCircuitTester`, published at ITiCSE in 2008. This tool allowed students and instructors to write unit tests for the simulated digital logic circuits

(adders, ALUs, CPUs, etc.). Before `JLSCircuitTester` was available, the only way to test circuits was to tediously enter various inputs and manually verify the results. Students would often spend too little time testing, and, as a result, submit buggy circuits. Worse yet, it would take me several days to test their submissions and provide feedback. By the time students did receive feedback, the class had moved onto other topics, so students rarely took the time to address the misconceptions and/or mistakes that led to bugs.

Once `JLSCircuitTester` was available, students could, in theory, write tests before beginning to build the circuit, which allows them to plan ahead and avoid having to make awkward "patches" at the end to fix bugs. The tool also allowed students to quickly and easily verify that their circuits passed their tests, so they were much less likely to submit buggy circuits. Furthermore, if they did submit a buggy circuit, I could quickly detect the bugs and return the circuit for correction within 24 hours. Introducing `JLSCircuitTester` raised the quality of project submissions significantly.

- Z. Kurmas. Improving student performance using automated testing of simulated digital logic circuits. *Proceedings of the 13th annual conference on Innovation and technology in computer science education (ITiCSE 2008),* July 2008.

MIPSUnit

I had similar problems with assembly language assignments as I did with digital logic (poor testing and code that had a poor design because bug fixes were "patched" on at the end), so I wrote `MIPSUnit`, a set of two unit testing frameworks for MIPS assembly. It was published at SIGCSE in 2017. Of the two frameworks, `MUnit` has been significantly more popular, so I will focus only on it.

`MUnit` (https://kurmasgvsu.github.io/Software/mipsunit_munit/) takes a fundamentally different approach to defining tests than `JLSCircuitTester`. Students found `JLSCircuitTester`'s custom syntax for defining tests very difficult to use and also found the output difficult to interpret. Students didn't tend to leverage the Test Driven Development opportunities. (They did benefit significantly from the quick feedback and opportunity to revise and resubmit.)

In response, I based `MUnit` on `JUnit`. Specifically, `MUnit` uses `JUnit`'s test-running engine and assert system. The tests themselves are calls to an API that (1) defines the initial MIPS CPU state, (2) simulates the code under test, and (3) provides access to the final CPU state for use in assertions. The output is simply the output generated by `JUnit`, with which our students were exposed to in CIS 162 and 163.

`MUnit` has been very successful. It has significantly improved the quality of submissions in my courses, and has been adopted and acknowledged by instructors at other universities. (See attached letter by James Moscala.) Not only that, but Andrew Peterson at the University of Toronto Mississauga found the design helpful enough that he and his students took the time to add RISC-V support. (See his attached letter.)

- Z. Kurmas. MIPSUnit: A Unit Testing Framework for MIPS Assembly. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education,* March, 2017.

DLUnit

Seeing students hesitate to write their own tests for circuits, and seeing the external success of `MUnit` motivated me to create `DLUnit`: a unit test framework for simulated digital logic circuits

that is based on `JUnit` in a similar manner to `MUnit`. In addition, `DLUnit` supports multiple digital logic simulators, which will make it appealing to more instructors. (`JLSCircuitTester` only supported the JLS simulator. `DLUnit` also supports the much more popular `Logisim` family of simulators.) https://kurmasgvsu.github.io/Software/DLUnit/

- Z. Kurmas. DLUnit: A Unit Testing Framework for Simulated Digital Logic Circuits. *Proceedings of the 2023 IEEE/ASEE Frontiers in Education Conference (FIE),* October, 2023.

Overall, the progression from `JLSCircuitTester`, to `MUnit`, to `DLUnit` represents the progression of an ongoing line of applied scholarship that represents my response to (1) the needs/preferences of the students, (2) ideas and preferences of colleagues at other universities, (3) trends in instruction, and (4) improvements in my own software development skills (partialy as a result of my sabbaticals).

ICOS

One of the challenges in teaching computer architecture today is that it is difficult, if not impossible to isolate the effects of individual CPU components (branch predictor, cache memory, etc.). As a result, the course can feel frustratingly "theoretical" to students. (I find myself saying things that reduce to "The CPU has a branch predictor. It makes programs faster. Trust me.") This challenge is compounded by the fact that things like threads, interrupts, and context switches can make the assembly code students see in class feel "distant" from the hardware being studied.

ICOS (https://github.com/kurmasz/ICOS) is a framework that allows students to run C code directly on a machine's "bare metal". It does not have enough features to be called a true operating system; but, it does (1) provide a simple boot loader, (2) load code from a USB drive, and (3) provide simple BIOS-based text output. This is just enough functionality to allow student to write and measure code without the overhead of a traditional operating system. It also allows students to get a close look at what happens "behind the scenes" as a computer boots up.

- Z. Kurmas. ICOS: Support for "Bare Metal" Computer Architecture Assignments *Proceedings of the 19th Workshop on Computer Architecture Education,* June 2017.

Community Finding / "Traditional" Research

I am a math nerd at heart (who just happens to love programming). I especially enjoy collaborating with others when their work involves interesting discrete math problems. For example, I contributed to Jerry Scripps's work on community detection/finding in social networks by looking into the benefits of using brute force to enumerate and evaluate all possible communities. (This work also aligned well with Christian Trefftz's interest in parallel computing.) My main contributions were to (1) develop a formula to count the number of possible communities in a given situation (e.g., to compute the size of the "search space") and (2) to incorporate Hugh McGuire's iteration algorithm into a framework that could be used to study various quality metrics for communities.

My work was a significant portion of one main paper and provided ongoing minor contributions to several "follow-up" papers.[5]

- Z. Kurmas, H. McGuire, J. Scripps, and C. Trefftz. (2014) Enumerating Communities for a Deeper Understanding of Community Finding. *IEEE/WIC/ACM International Joint Conferences on Web*
- Trefftz, C., Kurmas, Z. A., & Scripps, J. A. (2017). Covers, partitions and a heuristic to calculate two-covers. IEEE.
- Trefftz, C., Scripps, J. A., & Kurmas, Z. A. (2017). An introduction to elements of parallel programming with JAVA streams and/or thrust in a data structures and algorithms course. In John Meinke (Ed.), (1st ed., vol. 33, pp. 11-23). Journal of Computing Sciences in Colleges.
- Trefftz, C., McGuire, H. W., Kurmas, Z. A., Scripps, J. A., & Pineda, J. D. (2018). Exhaustive community enumeration on a cluster. *Proceedings of 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE.
- Scripps, J. A., Trefftz, C., & Kurmas, Z. A. (2018). The Difference between Optimal and Germane Communities. *Social Network Analysis and Mining / Springer, 8*(1), 44
- Cao, X., Scripps, J. A., Trefftz, C., & Kurmas, Z. A. (2019). Data Placement Strategy in Data Center. The 19th Annual IEEE International Conference on Electro Information Technology (EIT2019).

Other Teaching-Related Scholarly Activity

Industry Placement Sabbaticals

For both of my sabbaticals, I spent a year working in industry. (I included the details in the "Teaching" section because the description provided context needed there.)

Participation in NSF grants

I have participated in two NSF grants. (I helped execute these grants; I did not participate in writing them.)

- I helped Zack Butler and Ivona Bezakowa at the Rochester Institute of Technology (RIT) collect data for their "Pencil Puzzle" NSF grant. This grant studied the benefits of teaching introductory Computer Science principles using puzzles like Sudoku that can be completed using pencil and paper. My role was to assign both "regular" and pencil-puzzle-based assignments in several courses (CIS 163, CIS 263, CIS 371). At the end of the semester, I administered a survey to the students, and also participated in an interview with the project's PIs. See this paper for more details[6]:
  - Z. Butler, I. Bezáková, et al. Putting a Context in Context: Investigating the Context of Pencil Puzzles in Multiple Academic Environments**.** SIGCSE 2023: Proceedings of the 54th ACM Technical Symposium on Computer Science Education
- I am currently working with Suzanne Matthews at West Point and Kevin Webb at Swarthmore to write Runestone-based exercises for their online textbook Dive Into Systems.[7] This project began in early 2023.

KielceRB

`KielceRB` (https://github.com/kurmasz/KielceRB) is a highly customizable templating engine for generating assignments, syllabi, web pages and other course documents. It loads a hierarchy of key-

[5] The "main" 2014 paper is listed for context. Other than that, only papers from 2017 or later are listed here.
[6] Listed for context only. I am *not* a co-author.
[7] Again, listed/linked for context only.

value pairs from files at various file system levels. These values can then be inserted into documents using Ruby's ERB templating engine. `KielceRB` simplifies the maintenance of course documents by moving data that changes regularly into external data files where they can be easily identified and updated. For example, I can update my syllabus each semester by editing a data file containing only key dates, rather than having to search all my syllabus document for all the dates that change each semester.

Loading data from various file system levels makes it easy to share values among all documents for a particular course and/or semester. For example, I can put my contact info in a file at the top of my file system hierarchy and all course documents automatically have access. I need not maintain that information in several different places/documents. `KielceRB` also provides methods for including one document inside another allowing users to easily share common content among several pages (navigation bars, contact information, assignment headers, etc.).

Using `KielceRB` I can prepare the course web pages for an entire semester in less than an hour.

This tool was presented as a peer-reviewed paper at CCSC:Midwest in 2020:

- Kurmas, Z. A. (2020). KielceRB: A Highly Customizable Templating Engine for Course Documents. (5th ed., vol. 36). Evansville, IN: Journal of Computing Sciences in Colleges.

Powrawa

`Poprawa` (Polish for "improvement") is a Ruby library for generating progress reports from Excel-based gradebooks. It is designed for instructors who use proficiency grading (or other forms of alternative grading) but find that their LMS's gradebook doesn't meet their needs. (Yes, I'm looking at you, Blackboard.) See https://shorturl.at/tvwA2 for one example of how it can be difficult to track grades using an LMS.

This tool goes beyond a simple, personalized script: It is available on GitHub (https://github.com/kurmasz/poprawa). It is well-documented, and well-tested. It is motivated by my needs but is designed to solve a more general problem. It is usable (I've been using it for two semesters) but is not quite ready to be promoted widely.

Anna Carvalho worked on part of this library and presented her portion at Student Scholars Day in 2023.

Qualtrics

In 2020, I helped Carmen Fernandez-Florez and her team modify a Qualtrics module for her Implicit Association research. The Implicit Association Test (IAT) is widely used in psychology for detecting unconscious bias. Tom Carpenter et. al wrote a Qualtrics module to analyze implicit bias using associations between words and pictures (https://github.com/iatgen/iatgen). My role was to modify this module to use words and *sounds* instead of pictures (https://github.com/KurmasGVSU/iatgen). The purpose of Florez's research was to detect whether respondents held an unconscious bias based on the way a person pronounces certain sounds in Spanish. (In other words, the researchers are looking for bias based on accent.)

PLNQ

PLNQ (PrairieLearn Notebook Question) is a tool that simplifies the process of generating auto-graded Jupyter notebook-based programming assignments. This tool makes it much easier to prepare "Codingbat-like" practice for students in courses like 160, 161, 162, and 500.

This tool was accepted as a peer-reviewed poster presented at SIGCSE 2025 and is under review by the Journal of Open Source Education.

Source and documentation: https://github.com/kurmasz/plnq
PyPI: https://pypi.org/project/plnq-gvsu/

External Peer Appraisals (R9)

See attached letters from James Moscola and Andrew Peterson.

Service

Service Statement (S1)

I make a point of choosing service opportunities that align with my aptitude. I have found that my time is best utilized on policy-making committees such as the Faculty Salary and Budget Committee (FSBC) because

- I enjoy analyzing data (salary information, health care costs, etc.), and
- I enjoy the challenge of crafting policy that ensures that all faculty are treated fairly while still respecting the differences in culture and need among units.

My main areas of service over the past several years have been FSBC and the School's Student Affairs Committee. I have found that these two committees are the best uses of both my aptitude and student-focused approach to serving GVSU. This past year, I have also served on the COACHE Steering Committee, which is also well-aligned with my service goals.

University Service (S2)

FSBC

I have served on the Faculty Salary and Budget Committee (FSBC) since at least 2008. FSBC is not just a "sit-and-listen" committee. I have served on several *significant* ad-hoc sub-committees including (a) a sub-committee that collected data in response to Provost Davis's proposal to significantly limit the number of sabbaticals allowed each year, and (b) a sub-committee to address the limitations of our annual salary-increase program. The sabbatical sub-committee's quiet, but *lasting contribution* was that the Provost discretely and unofficially dropped her hard limit on the number of sabbaticals. (See the letters of support from Lawrence Burns and Robert Hollister for details on these subcommittees.)

FSBC meets for 90 minutes every other Friday. This page contains the archive of FSBC annual newsletters: https://www.gvsu.edu/fsbc/newsletters-supplemental-materials-faculty-log-in-required-12.htm

Student Affairs

My desire to choose activities that directly affect students has also led me to serve on the Student Affairs committee. I was the unofficial chair of the committee from about 2019 until the committee disbanded with the formation of the new College. In addition to the baseline work of making scholarship and award recommendations, we rewrote the essay prompts for our various scholarships so that applicants will write essays that more directly address the qualities that the donors are looking for.

COACHE Steering Committee

During Winter semester, faculty were asked to complete the COACHE satisfaction survey. This survey showed several areas of concern, including significant weaknesses in the relationship between faculty and administration. The main task of the COACHE Steering Committee was to develop an action plan to address the areas of concern. This May, we brought in Susan Elrod and Mary Papazian to lead two 2-day workshops in Change Leadership, which taught participants how to identify the most significant areas of

concern, then partner with leadership to address those concerns over the long term. Our task going forward is to now apply the techniques we've learned to the concerns raised and see that we follow through with the proposed changes.

Given the number of dissatisfied faculty university-wide, as well as the level of dissatisfaction expressed by several of Grand Valley's most productive and influential faculty (both inside and outside the College of Computing), my work on the COACHE Steering Committee is my most significant contribution of 2024 and 2025. It may end up being my most significant service contribution while at GVSU.

Other Committees

I have at times also been a member of the Faculty Personnel Policy Committee (FPPC), the Faculty Grievance Committee, the University Conduct Committee, and the College Professional Development Committee.

Community Service (S3)

Candi and I participated in Freshman Move-In from 2004 through 2018.

I have volunteered with Science Olympiad in several roles since 2021. Most of my work has been

- Helping the Allendale Science Olympiad team when necessary
- Preparing and/or proctoring exams for the CodeBusters event.

My most significant activity was to serve as the CodeBusters Event Supervisor for the State of Michigan competition in May 2024.

Other (S3)

Study Abroad

In 2019, I worked with Tadeusz Wilusz of the Academy of Economics in Kraków (AE) to provide an opportunity for our students to complete their internship abroad. My goal was to address the two most common concerns I hear from students about studying abroad: (1) "It costs too much," and (2) "I need/want to complete my internship during the Summer." The Academy offered to host two students for the summer, providing them room and board, and an opportunity to assist in their IT department. The students would not be paid for their work (which is difficult given Polish laws regarding foreign workers); but, the free room and board would have effectively allowed students to spend the summer abroad in Poland and complete their internship at a cost not much higher than that of a plane ticket, tuition, and incidentals.

The agreement was signed in early 2020, just before Covid; so, no students have yet participated. I have contacted Ehren Kuzekov for an update on the status of this agreement.

Misc.

I have also engaged in "baseline" service including Convocation, Commencement, participating in job searches/interviews, advising, etc.

Index of Dissemination

Important peer-reviewed papers before 2017 (for context):

- Z. Kurmas, H. McGuire, J. Scripps, and C. Trefftz. Enumerating Communities for a Deeper Understanding of Community Finding. *IEEE/WIC/ACM International Joint Conferences on Web,* August 2014.
- Z. Kurmas. Improving student performance using automated testing of simulated digital logic circuits. *Proceedings of the 13th annual conference on Innovation and technology in computer science education (ITiCSE 2008),* July 2008.

Primary[8] peer-reviewed papers (2017-present):

- Z. Kurmas. DLUnit: A Unit Testing Framework for Simulated Digital Logic Circuits. *Proceedings of the 2023 IEEE/ASEE Frontiers in Education Conference (FIE),* October, 2023.
- E. Spertus, Z. Kurmas. Mastery-Based Learning in Undergraduate Computer Architecture, Proceedings of the 21st Workshop on Computer Architecture Education, June 2021.
- Kurmas, Z. A. (2020). KielceRB: A Highly Customizable Templating Engine for Course Documents. (5th ed., vol. 36). Evansville, IN: Journal of Computing Sciences in Colleges.
- Z. Kurmas. ICOS: Support for "Bare Metal" Computer Architecture Assignments *Proceedings of the 19th Workshop on Computer Architecture Education,* June 2017.
- Z. Kurmas. MIPSUnit: A Unit Testing Framework for MIPS Assembly. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education,* March, 2017.

Secondary[9] peer-reviewed papers (2017-present):

- Cao, X., Scripps, J. A., Trefftz, C., & Kurmas, Z. A. Data Placement Strategy in Data Center. The 19th Annual IEEE International Conference on Electro Information Technology. 2019
- Scripps, J. A., Trefftz, C., & Kurmas, Z. A. The Difference between Optimal and Germane Communities. *Social Network Analysis and Mining / Springer, 8*(1), 44. 2018
- Trefftz, C., McGuire, H. W., Kurmas, Z. A., Scripps, J. A., & Pineda, J. D. (2018). Exhaustive community enumeration on a cluster. *Proceedings of 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE. 2018
- Trefftz, C., Scripps, J. A., & Kurmas, Z. A. An introduction to elements of parallel programming with JAVA streams and/or thrust in a data structures and algorithms course. In John Meinke (Ed.), (1st ed., vol. 33, pp. 11-23). Journal of Computing Sciences in Colleges. 2017

Peer-reviewed presentations (panels, tutorials, workshops, etc.)

- CCSC:MW tutorial on automated testing of student assignments using GitHub Classroom and GitHub Actions (motivated by my experiences with CI/CD during my last sabbatical). 2021
- SIGCSE Panel: R McFall, Z. Kurmas P. Conrad D. Frailey. Why and How to Spend a Sabbatical in Industry. SIGCSE '19: Proceedings of the 50th ACM Technical Symposium on Computer Science Education. February 2019
- SIGCSE Workshop[10]: Z. Kurmas. Testing Across the Curriculum. Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education. 2017

[8] These are either single-author papers, or papers where I did a minimum of 50% of the work.
[9] These are co-authored papers where I did less than 25% of the work.
[10] This workshop was accepted (i.e., passed peer-review) but was one of several that year canceled due to low enrollment.

Software

In many cases, the most important result of my scholarly activity is not the formal publication, but the software artifacts that are available for students and faculty to use:

DLUnit	A unit testing framework for simulated digital logic circuits
MIPSUnit	A unit testing framework for MIPS assembly
ICOS	A toolkit for running experiments on "bare metal" (in Computer Architecture courses)
KielceRB	A templating system for maintaining assignments and other course documents.
Ghc.rb	A command-line tool for interacting with GitHub Classroom
Poprawa	A library for generating progress reports (for use in courses that use mastery grading)