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 have the largest benefit for our students.

Highlights

Below are brief descriptions of my most important contributions.

Teaching Summary

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, as discussed below, I make a point of choosing scholarship, service, and sabbatical projects that directly benefit students.

Because of these efforts, I was nominated for the GVSU Outstanding Teacher Award in 2022.

In addition to simply teaching courses, I have been the thought leader / promoter / early adopter of three important changes:

1. Making unit testing an integral part of several courses (including 162, 163, and 351)
2. Using GitHub Classroom and GitHub Actions to automate the checking of students' assignments
3. Adopting principles from "Grading for Growth"[1]

Scholarship Summary

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. My most significant projects have been those that produce software for students to use. 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)

[1] I've often used the term "mastery grading" to describe the changes I've been making. But, that description is both incomplete and misleading. I'm working to get into the habit of saying "proficiency-based grading," which is better, but still not ideal.

`MIPSUnit` was accepted as 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

Service Summary

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.

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 approved each year, and (b) a sub-committee to address the limitations of our annual salary-increase program.

I have included letters from Lawrence Burns and Robert Hollister discussing my contributions to FSBC.

My desire to choose activities that directly affect students has also led me to serve on the Student Affairs committee. In addition to the baseline work of making scholarship and award recommendations, we are working to improve 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. As of June, 2023, we have revised the prompts for the Maino and Gillette scholarships.

Details

This section provides the details of my most significant teaching, scholarship, and service activites.

Teaching

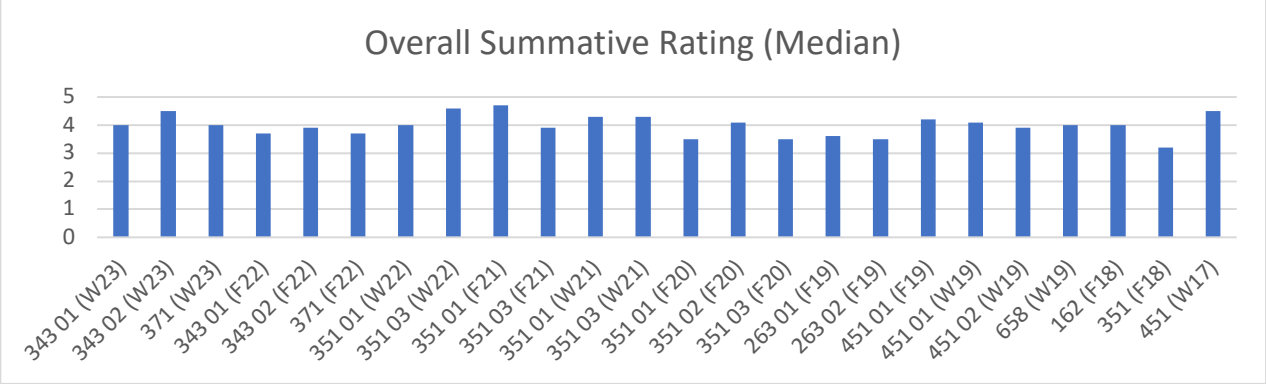
> Full Professor - seven full time equivalent years at the rank of an Associate Professor is assumed along with demonstrated *excellence in teaching*. In addition, the faculty member must have engaged in several meritorious activities and accomplishments that extend *beyond normal teaching duties* and performance during the previous six years.

Excellence in Teaching

I make a point of maintaining high expectations for students, but also providing the necessary support to help students meet those expectations. The evidence below clearly demonstrates that my teaching meets the expectations for promotion.

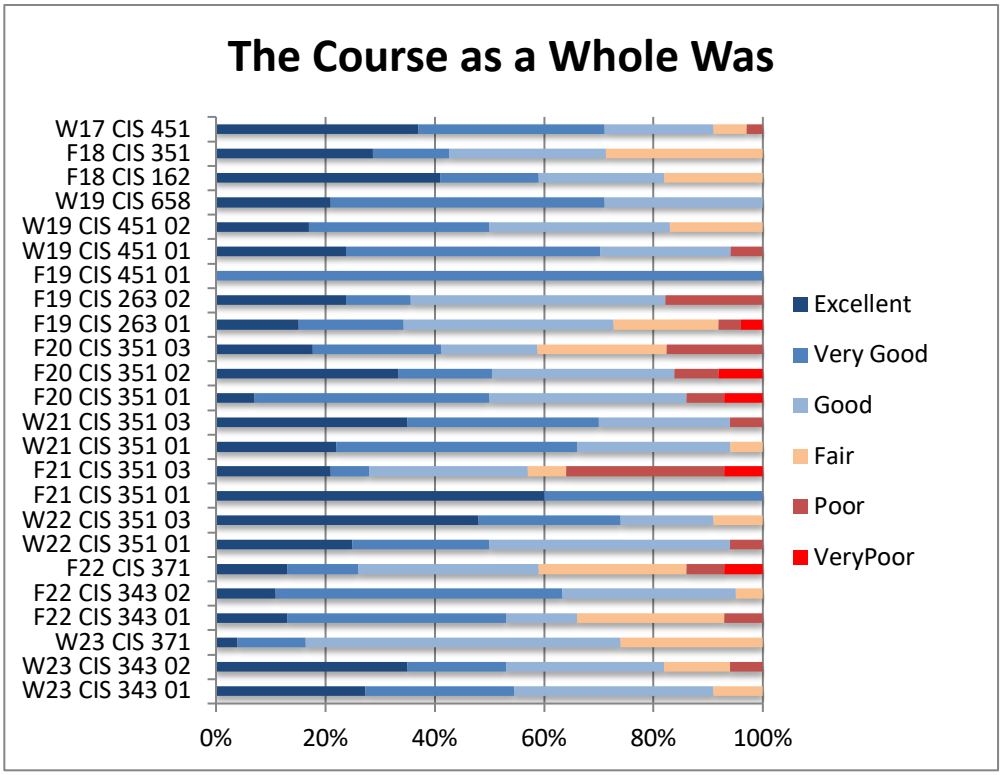
Student Evaluations

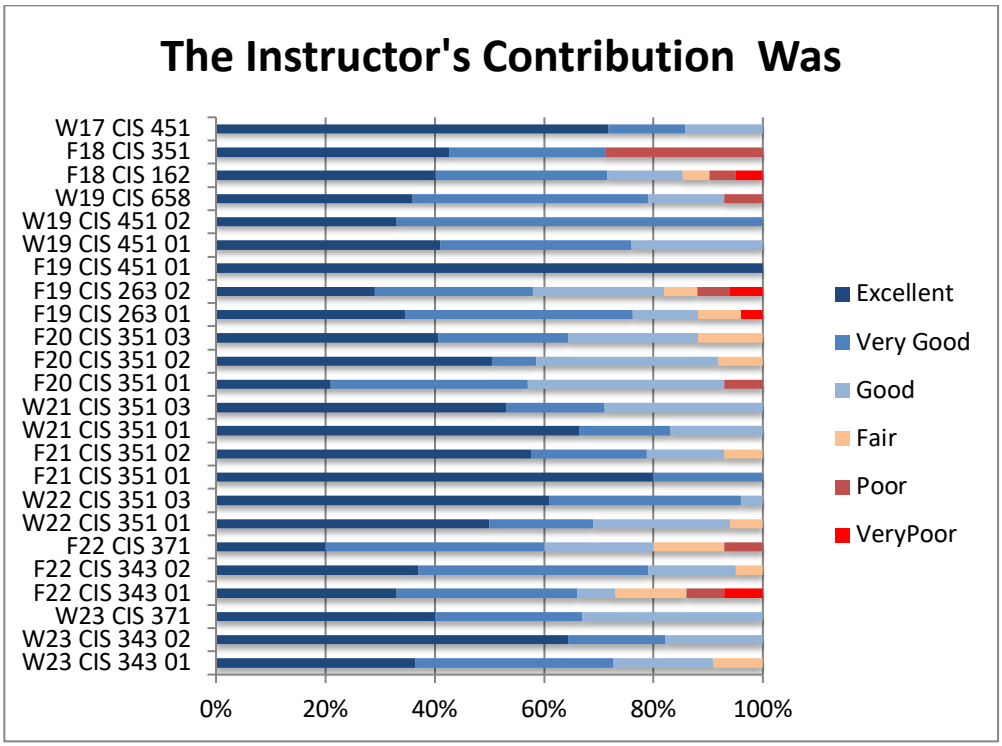
The "overall summative rating" of my LIFT evaluations are often at or above 4.0 and, as can be seen below, have fallen below 3.5 only once since 2017. Lower scores tend to be correlated with new preps and/or big changes like my switch to Proficiency-based grading.[2]



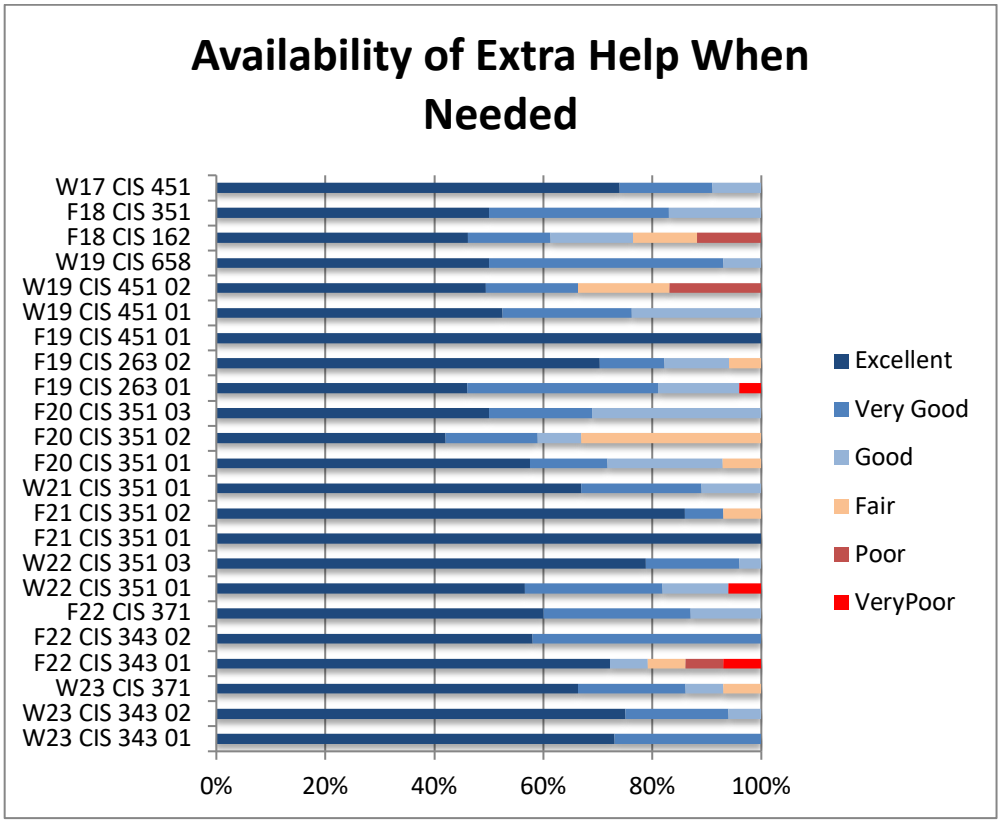
[2] Thanks to Jonathan Leidig for sharing the tools to prepare these charts.

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 14 out of 24 sections since 2017, at least 80% of students responded with “Excellent” or “Very Good” when asked whether “Extra help was readily available.” That figure increases to 20 out of 24 sections when including students who selected “Good”.



Nominated for University Outstanding Teacher Award

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

Beyond Normal Teaching Duties

I have engaged in several activities that “*extend beyond the normal teaching duties*:”

Industry Placement Sabbaticals

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

Note: The sabbaticals themselves are professional development, not teaching. I discuss them here because they provide important context for the sections that follow.

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 the CIS 162 lab). 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 this SIGCSE panel with Ryan McFall:

- 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

"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 students feedback 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 (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 these tools. Others have since adopted similar processes. Some have followed my

[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 has simplified that process significantly.

lead by adopting "professional" tools like GitHub; others have adopted about 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)

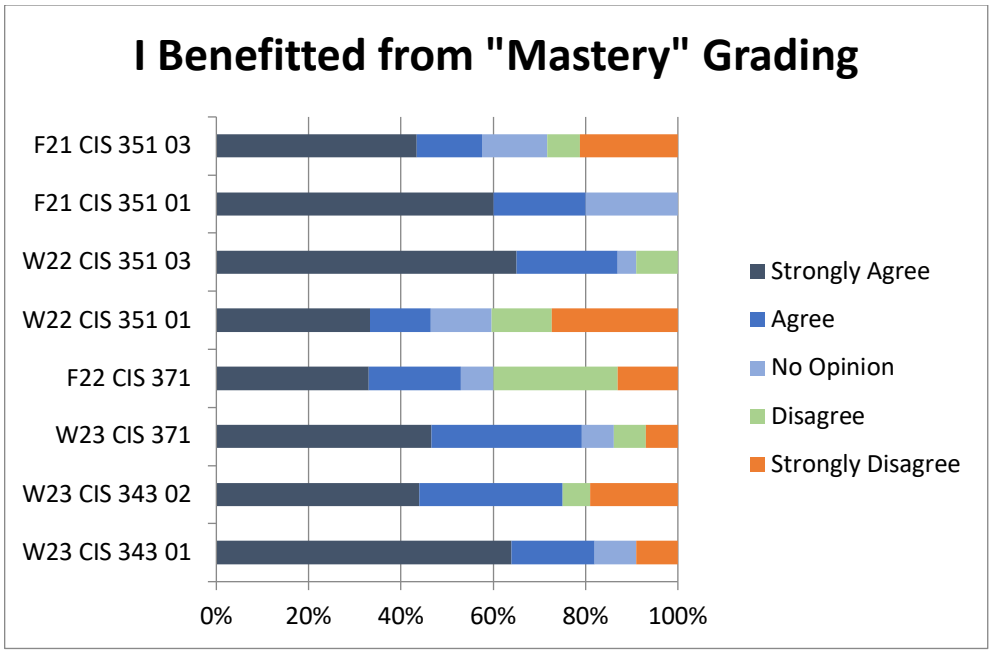
Adopting principles from "Grading for Growth"

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.[5] 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-based grading".) I made mistakes the first few semesters; and there is still a lot of room for improvement; but, overall, a "Grading for Growth" approach is clearly more effective than traditional grading for *most* students.

I added the following question to my LIFT evals: "I benefitted from the 'Mastery Grading' approach to this course". The chart below shows that a clear majority of students believe they benefit from this approach. In addition, work submitted on tests and quizzes as well as conversations with students during office hours strongly suggest that most students are learning better. ("Believe" and "suggest" are the key words above. The only thing I can formally claim is that students can more effectively communicate what they've learned. But, even that alone is an important benefit.)

[5] Their book just came out: https://www.routledge.com/Grading-for-Growth-A-Guide-to-Alternative-Grading-Practices-that-Promote/Clark-Talbert/p/book/9781642673814



The downside is that about 20% of students don't believe they benefit from my implementation of proficiency grading. It appears that the main problem for many of these students is the "snowball effect": A student who fails to meet an assignment's expectations must then take time to both address the deficiency of that assignment and keep up with new material. Previous conversations with others (and recent conversations with School of Computing colleagues) both suggest that adjusting my D/F line will reduce the stress students feel when they struggle with a topic and/or need to make revisions. My focus this summer has been to set my D/F line appropriately for fall courses.

It is too soon to tell what effect my efforts in "Grading for Growth" will have on the School of Computing; but, given how successful my approach has been in my classroom for the majority of students, 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 --- especially once I figure out how to mitigate the "snowball effect".

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, 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/playlists). I'd love to make more; however, making videos of this quality simply takes too much time --- especially

considering there don't appear to be any opportunities for peer-reviewed dissemination in this space.

Scholarship

> Full Professor - ***acknowledged*** professional recognition through scholarship and creative activity is expected. In addition to meeting the expectation for Associate Professors, Professors must demonstrate ***ongoing*** scholarly accomplishments.

Automated Testing / Feedback (Acknowledged and ongoing)

The main theme of my *ongoing* 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.

JLSCircuitTester (context)

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 that and return it 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` 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.)

This new tool has made a big improvement in my courses. It is just now being published, so it is too early to tell how popular it will become outside of GVSU.

- 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 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 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.[6]

- 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.)

[6] The "main" 2014 paper is listed for context. Other than that, only papers from 2017 or later are listed here.

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[7]:
  - 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.[8] This project began in early 2023.

KielceRB

`KielceRB` is a highly customizable templating engine for generating assignments, syllabi, web pages and other course documents. It loads a hierarchy of key-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-based 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.

---

[7] Listed for context only. I am *not* a co-author.
[8] Again, listed/linked for context only.

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.)

Prof. Florez used this modified module to conduct a preliminary study and presented it at a conference (which doesn't have the same level of rigor and peer-review as a conference in our discipline). She expects to have a formal, rigorous, peer-reviewed publication later this year.

Service

> Full Professor - A *sustained* record of service *within* and *beyond* the unit is expected. Full professors demonstrate *leadership* measured in *significant and lasting contributions* to the unit and beyond (college, university, profession, or community).

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.

FSBC

I have served on the Faculty Salary and Budget Committee (FSBC) since at least 2008 ("*sustained*"). I especially enjoyed serving on this committee 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.

FSBC is not just a "sit-and-listen" committee. I have served on several *significant* 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.)

Student Affairs

My desire to choose activities that directly affect students has also led me to serve on the Student Affairs committee. I have been the unofficial *leader* of the committee since about 2019. In addition to the baseline work of making scholarship and award recommendations, we are working to improve 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. As of June, 2023, we have revised the prompts for the Maino and Gillette scholarships.

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 to see what the status of this agreement is.

Misc.

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.

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

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[9] 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.

[9] These are either single-author papers, or papers where I did a minimum of 50% of the work.

- 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[10] 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[11]: Z. Kurmas. Testing Across the Curriculum. Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education. 2017

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:

`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)

[10] These are co-authored papers where I did less than 25% of the work.
[11] Note: This workshop was accepted (i.e., passed peer-review) but was one of several that year canceled due to low enrollment.