CSE 398/498 Software Performance Engineering

Course Description (Catalog Version)

Principles and techniques for optimizing the performance of software. Topics include performance evaluation, algorithmic optimizations, cache behaviors, impact of microarchitecture on software implementation, compiler optimizations, vectorization, and parallelization.

Course Description (Website Version)

As computing continues to grow in scale and scope, the consequences of poor performance become increasingly drastic. As a simple example, consider how a single workload running on a top-10 high-performance computing cluster can consume millions of core hours every hour, or how a database or AI workload in a datacenter can consume more than a billion core hours per day. If such a program could be sped up by just five percent, the savings, in terms of energy, e-waste, system administration time, and datacenter infrastructure would be tremendous.

This class provides students with an opportunity to explore the sources of slowdown in modern software, and learn techniques for improving performance. The focus is intentionally broad, considering everything from how to rewrite code based on an understanding of a specific workload down to generalizable techniques to optimize low-level interactions with hardware, including storage, network, and memory. The class will include several programming assignments, through which students will learn the basics of measuring and improving performance. The culminating experience in the class will be a longer-term project in which students conduct a deep dive on an optimization technique and present their end-to-end experience.

Locations and Times

- Class Meetings
  - 11:15 AM -- 12:30 PM M/W
  - BC 115
- Office Hours
  - 12:30 PM -- 2:00 PM M/W
  - BC 339

Additional appointments for office hours can be arranged as needed.

Personnel / Contact Information

- Prof. Michael Spear
  - Office: BC 339
  - Email: mfs409@lehigh.edu (note: all other emails you find online will forward to this account)
  - Phone: (610) 758-3285
  - Emergency Contact: (484) 619-6813

Course Links

CourseSite will be used for one-way communication from me to the students. I'll use it to share papers, slides, and other material.

GitLab will be used for two-way communication of graded work. I'll use it to share per-student starter code, and to collect student work. I will create and manage a private repository for each student. Please **do not fork**

the repository I provide.

Piazza will be used for offline and asynchronous discussion. Piazza is configured for fully anonymous posting, as well as private (student-instructor) posts, so please favor it for course-related communications, unless you are discussing sensitive issues that are better handled in-person or via email.

Learning Outcomes

Students will learn to:

- Understand the sources of overhead in modern software
- Measure and evaluate software behaviors to identify performance anomalies and develop hypotheses to explain their root causes
- Design and develop optimizations that preserve functional requirements while increasing software performance

Health and Safety

This class will only be fun and rewarding if everyone shows up and participates. To that end, attendance is mandatory, and I will deduct one course point for each unexcused absence.

With that said, I understand that there are many reasons why a student might need to miss class. As long as you coordinate with me before class (even by sending an email 5 minutes before class), I will treat your absence as excused. If you are feeling ill, please **do not** send a doctor's note. If you have a different reason, whatever it is, please **do not lie and say you are sick**. Just tell me the real reason. My default will be to excuse your absence.

If I feel that a student is abusing this policy, I reserve the right to revoke it on a student-by-student basis.

Weekly Schedule of Topics

In no particular order, I think we should discuss the following topics this semester. On the first day, we will discuss these in more detail and come up with an ordering. Students may also propose other topics.

- Diagnosing Performance Bugs
- How To Measure Performance
- How To Present Performance Results
- Manual Instruction Selection
- Network System Optimization
- Optimizing For Specific Workloads
- Profile-Guided Optimization
- Reducing To A Problem With An Efficient Solution
- Scheduling Before Executing
- Storage System Optimization
- The Impact Of The Memory System
- The Importance of Scale
- The Role Of The Compiler
- The Role Of The Run-Time Environment
- Threading / Concurrency / Parallelism
- Vectorization

Once we've settled on the topics, I will update this table:

Date	Topic
Jan 19/21	
Jan 26/28	
Feb 2/4	
Feb 9/11	
Feb 16/18	
Feb 23/25	
Mar 2/4	
Mar 9/11	Spring Break -- No Class
Mar 16/18	
Mar 23/25	
Mar 30/Apr 1	
Apr 6/8	
Apr 13/15	
Apr 20/22	
Apr 27/29	Final Project Presentations

Graded Assignments

There will be three programming assignments and a semester project. The due dates for these assignments are as follows (tentative):

Assignment	Due Date
Discussion Leader Sign-Up	Jan 23
Program #1	Feb 4
Program #2	Feb 25
Semester Project Proposal	Mar 4
Program #4	Mar 18
Semester Project Presentation	Apr 27/29
Semester Project One-On-One	May 4--8
Discussion Leader	Varies

Final grades will be assigned according to the following distribution:

A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F
100-92	91-90	89-87	86-82	81-80	79-77	76-72	71-70	69-67	66-62	61-60	59-0

There will not be a curve in this class. It is possible for every student to earn an A.

No late assignments will be accepted, unless an extension is coordinated at least 48 hours before the assignment is due.

Programming Assignment Details

The programming assignments will be individual tasks for students at the 400 level. Students who registered for the course at the 300 level are allowed to work in teams of 2. They will be due at 11:59 PM on the due date, via Git. The exact assignments are up for discussion. Initially, these are the projects under consideration:

Efficient Sorting Algorithms

The goal of this assignment is to get a survey of several key concepts in the class. Students will compete to produce the fastest implementation of a sorting algorithm. The assignment will be worth 15% of the course grade. Concepts in this assignment include:

- Fine grained time measurement
- Hybrid algorithms
- Using hardware performance counters
- Impact of the workload on algorithm performance (including data set size, data types, and initial input state)
- Role of locality
- SIMD and parallelization
- Optimizations for I/O

Data Structure Microbenchmarking

This is another assignment that surveys key concepts in the class. Students will compete to produce fast implementations of ordered and unordered map data structures. The assignment will be worth 15% of the course grade. Concepts in the assignment include:

- Memory hierarchy issues
- The role of randomness
- The impact of variability in workload (including data set size, key and value types, and operation ratios)
- Concurrency and multithreading issues

Sparse and Dense Matrix Multiplication

This assignment places a heavier emphasis on analyzing the input and dynamically choosing an algorithm for multiplying a set of matrices. Of course, many key concepts in the class will matter. Again, students will compete to produce the fastest implementations. The assignment will be worth 15% of the course grade. Concepts in this assignment include:

- Predicting workload behavior
- Redesigning data structures and algorithms around data properties
- Caching
- Parallelism

Semester Project

Students will propose a project that allows them to demonstrate significant skill in software performance engineering. This will require students to identify a problem that suffers from a significant performance bug and then solve it. Students may work in teams of 3 for this assignment. This assignment will be worth 45% of the course grade.

Students must determine their problem and have it approved by the professor by March 2nd. They will be expected to give a 5 minute presentation about their project on March 4th. This presentation will be worth 20% of the overall project grade, and is expected to demonstrate more than superficial effort (at a minimum, the target program has been run and measured).

Students are encouraged to use problems from their research and entrepreneurship as the foundation for their class projects. The professor has a list of projects that are available for students who are not able to determine a project on their own.

Final project results will be presented to the class in the final week of the semester. Each team will also schedule a discussion of their work for the first week of finals. Of course, students should be meeting with the professor regularly during the semester for guidance as they work on their projects.

Discussion Leader

Student teams are required to serve as the "discussion leader" for a topic/week of the class. Discussion leader teams are allowed to be different from teams for the project and assignments. Discussion leader responsibilities include proposing readings for the topic, guiding the in-class discussion, and producing on-line material that summarizes the classroom discussion. Discussion leader duties will be worth 10% of the course grade.

There are three purposes to the discussion leader responsibility.

- Encourage ownership of the material, by allowing students to direct the low-level details of a week's topic
- Facilitate in-class participation by reducing the need for all students to feel that they should be taking notes instead of joining in the discussion
- Develop an on-line repository of materials that can be shared with the broader Software Performance Engineering community, and that showcases students' technical communication skills

Discussion leaders' materials will be hosted on a public GitHub pages site, and work will be attributed to the scribes for any given week. Students must sign up for discussion leader responsibilities by January 23rd.

Academic Integrity

By this point in your academic career, you've seen this statement quite often:

> *All members of the Lehigh community have a responsibility to maintain academic integrity. Resources and details of expectations at Lehigh are available on the Provost’s website. It is expected that all students will abide by these standards throughout the course (e.g., homework, quizzes, papers, exams, projects, etc.). Academic integrity case studies will be discussed on the first day of class, and students are encouraged to ask questions for further clarity throughout the semester. Violations of academic integrity standards will not be tolerated and will be handled according to the guidelines in the University’s Student Conduct System.*

Here's what it means to me:

> *Someone is paying a lot of money for you to be here. It doesn't matter what people said in the past about the value of the Lehigh brand, or the value of the Lehigh alumni network -- your future depends on only two things: mastering the skill of doing deep thinking in your discipline, and mastering the skill of doing deep work in your discipline. If you cheat, you're wasting your money and my time. You're also hurting your classmates, because we are all connected, and when you don't know your stuff, and you embarrass yourself in a job interview, you taint the pool for everyone who comes after you. It's a prisoner's dilemma, and it's amplified by the fact that companies are orders of magnitude better at using big data than you are.*
>
> *I will try to give you the benefit of the doubt when it comes to academic integrity. After all, this is an elective course, so you should only be taking it because you think that really learning the material is worth it. However, if you violate my trust, I will have no tolerance.*
>
> *I ask that you are reasonable in terms of how you use online resources to help with your learning. Reading existing StackOverflow posts is wise. Asking questions on StackOverflow before asking them on Piazza is not. Reading blogs, research papers, and other on-line articles, and trying to understand them, is wise. Asking generative AI to summarize them is not. Using an IDE that helps you catch bugs in your code, and writing comments to help explain your code, is wise. Using generative AI to produce code that you do not understand is not. Be wise, because your learning, and hence your future, depend on it.*

Accommodations for Students with Disabilities

Lehigh University is committed to maintaining an equitable and inclusive community and welcomes students with disabilities into all of the University’s educational programs. In order to receive consideration for reasonable accommodations, a student with a disability must contact Disability Support Services (DSS), provide documentation, and participate in an interactive review process. If the documentation supports a request for reasonable accommodations, DSS will provide students with a Letter of Accommodations. Students who are approved for accommodations at Lehigh should share this letter and discuss their accommodations and learning needs with instructors as early in the semester as possible. For more information or to request services, please contact Disability Support Services in person in Williams Hall, Suite 301, via phone at 610-758-4152, via email at indss@lehigh.edu, or online.

The Principles of Our Equitable Community

Lehigh University endorses The Principles of Our Equitable Community. I expect each member of this class to acknowledge and practice these Principles. Respect for each other and for differing viewpoints is a vital component of the learning environment inside and outside the classroom.

Lehigh University Policy on Harassment and Non-Discrimination

Lehigh University upholds the Principles of Our Equitable Community and is committed to an educational, working, co-curricular, social, and living environment for faculty, staff, and students. The University does not discriminate in its admissions practices, employment practices, or educational programs or activities on the basis of age, color, disability, ethnicity, familial status, gender expression, gender identity, genetic information, marital status, national origin (including shared ancestry), pregnancy or related conditions, race, religion, sex, sexual orientation, and veteran or military status. Harassment or discrimination is unacceptable behavior and will not be tolerated. The University strongly encourages (and, depending upon the circumstances, may require) students, faculty, or staff who experience or witness harassment or discrimination, or have information about harassment or discrimination in University programs or activities, to immediately report such conduct.

If you have questions about Lehigh’s Policy on Harassment and Non-Discrimination or need to report harassment or discrimination, contact the Equal Opportunity Compliance Coordinator (Alumni Memorial

Building / 610 758 3535 / eocc@lehigh.edu).