CSCI 1675, Fall 2025: Designing High-Performance Network Systems

Role	Name	Office Hours
Instructor	Akshay Narayan akshayn@brown.edu	TBD
TAs	TBD	

Description

How can we understand the performance of network systems and make them faster? To explore this question, we will break it down into pieces, including (i) how to define and measure a system's performance, (ii) how to determine what factors in the system's design affect its performance, and (iii) how to use this information to make the system faster. Through lectures, readings, and hands-on programming projects, we will explore topics including throughput-latency curves, tradeoffs between open and closed request generation, and concurrency. Hands-on projects will involve modern Linux technologies such as perf and io_uring, as well as modern microservice architecture tools such as Kubernetes. The class will consist of lectures, short homework assignments that reinforce lecture content, and 4 project assignments with programming and technical writing components that explore these topics in greater detail.

Format

The course will consist of lectures with interactive discussion components, four hands-on implementation projects, and three homework assignments involving writing responses to readings.

Prerequisites

CS1670/2670, CS1680/2680, CS2690, CS1380, or graduate standing.

Components and Grading

Learning Goals

After taking this class, students will be able to measure, analyze, and improve the performance of a network system. Further, students will know the purpose of and be familiar with using the following tools, technologies, and APIs: perf, flamegraphs, vectored IO, io_uring, multitasking, container orchestration, and distributed tracing. Students will further know the purpose of, but may not be familiar with using the following tools: eBPF, RDMA, and DPDK.

Course Materials

There are no expected costs. All readings will be open-access. The course staff will provide computing resources to evaluate projects.

Attendance

Attending lectures is recommended (since lectures involve interactive components) but not required. Lectures will be recorded for later viewing.

Course Projects: 65%

The largest component of this class is a series of four course projects. Each project will involve an implementation task, an experimental evaluation task, and a project writeup. Course staff will grade projects based on the correctness of the implementation and experimental evaluation components, and the clarity and completeness of the project writeup. Projects will be graded during meetings with course staff, and students should be able to answer technical questions and defend their project's correctness usign quantitative evidence in their writeup during these grading meetings.

Rust

Projects in this offering of 1675 will be implemented in Rust. Our colleagues at Brown have recently published a great book to help students learn Rust; we expect that Chapter 4, "Ownership," will be especially helpful for this class. There are also great traditional resources such as The Rust Programming Language book, the standard library documentation, and Rust by Example.

Homeworks: 35%

This class will also involve reading-based homework assignments. Students will read a technical document that reinforces lecture content and answer questions that evaluate their understanding. Homework assignments will be graded on writing clarity and response correctness.

Time Breakdown

Students should expect to spend 3 hours per week in lecture, a further 3 hours per week on homework assignments, and approximately 100 hours on the class projects, for a total of at least 180 hours over the course of the semester.

Late Days

To conserve the course staff's grading effort, projects should be completed on time. However, in recognition of the need for some flexibility, students are allowed three "project late days" to submit projects past their deadline with no penalty, and one “homework late day” to submit homework assignments past their deadline with no penalty. Project late days cannot be used for homework assignments and vice versa. Late days cannot be used past the week the assignment is due in (i.e., the latest possible submission date for an assignment is at 23:59 on the Friday of the week it is originally due). Assignments won’t be accepted for grading beyond their last possible submission date or if no late days remain. There is no credit for unused late days at the end of the semester. Any fraction of a late day used counts as one late day. While weekends and University holidays don’t consume late days, note the above policy about the latest possible due date of each assignment. If serious extenuating circumstances arise after late days are exhausted, contact Akshay.

Collaboration and Academic Integrity Policy

Students should understand and follow the Brown Academic Code and the Code of Student Conduct.

Additionally, specific to this course, we encourage working with other students to build conceptual understanding and debug software issues. However, each student is responsible for their own project implementation and writeup, and their own homework response. Students must understand their submissions; instructors will interview and quiz students about their answers as part of the grading process to determine this.

For all class assignments, students must cite all sources (people, websites, papers, etc.) that they consult as a part of their work. External sources include but are not limited to previously published articles, blog posts, Stackoverflow or similar sites, conversations with other people, etc. This policy is not meant to discourage the use of external sources, but rather to codify a standard academic practice. Be generous with citations.

Finally, by taking this class, you agree to never post solutions for any assignments publicly.

Tentative Schedule

Key Dates

Dates are tentative. Changes will be announced in class.

Date	Event
TBD	HW 0: Evaluating Systems released
TBD	Project 0: Woonsocket released
TBD	HW 0 due
TBD	End of shopping period
TBD	Project 0 due

Date	Event
TBD	Project 1: Modern OS Interfaces released
TBD	HW 1: C10K released
TBD	Project 1 due
TBD	Project 2: Concurrency released
TBD	HW 1 due
TBD	Project 2 due
TBD	HW 2: Microservices released
TBD	Project 3: Microservices released
TBD	HW 2 due
TBD	Project 3 due

Quick Links

- Week 0
- Week 1
- Week 2
- Week 3
- Week 4
- Week 5
- Week 6
- Week 7
- Week 8
- Week 9
- Week 10
- Week 11
- Week 12
- Week 13

Week 0

Class intro.

Week 1

What is performance?

- Day 1: Throughput vs latency
- Day 2: Open-loop and closed-loop request generation

Week 2

Instrumentation

- Day 1: How do we measure performance? perf, flamegraphs
- Day 2: Memory hierarchy and data structures

Week 3

IO

- Day 1: Syscalls and IO, C10K problem, nonblocking APIs
- Day 2: Vectored IO and io_uring

Week 4

Kernel Extensions

- Day 1: University holiday. No class.
- Day 2: eBFP/XDP

Week 5

Kernel Bypass

- Day 1: RDMA
- Day 2: DPDK

Week 6

Concurrency

- Day 1: Threads and Coroutines, M:N model
- Day 2: Shared memory vs channels, Strong vs Weak scaling, Amdahl's law

Week 7

Scheduling

- Day 1: Preemptive vs Cooperative multitasking
- Day 2: Work stealing

Week 8

Isolation

- Day 1: VMs and containers
- Day 2: RPCs and microservices

Week 9

No class.

Week 10

Queueing Theory

- Day 1: Backpressure

- Day 2: Load Balancing

Week 11

Caching and Storage

- Day 1: Storage
- Day 2: Caching

Week 12

The Tail at Scale

- Day 1: Livelock and deadlock
- Day 2: Distributed tracing

Week 13

Cloud Computing

- Day 1: Cluster schedulers, stragglers
- Day 2: Resource Disaggregation

Registration

Due to constrained TA resources, this offering of CS 1675 will be capped. Students requesting an override must (a) have fulfilled the prerequisites and (b) complete course assignments during shopping period. All registered students will be responsible for completing all course assignments regardless of when they register for the class.

Auditing

If you wish to audit the class, note that the aforementioned limits on TA resources still apply, so we won't be able to offer auditors grading support or support in office hours.

Accessibility and Accommodations Statement

Brown University is committed to full inclusion of all students. Please inform me early in the term if you may require accommodations or modification of any of course procedures. You may speak with me after class, during office hours, or by appointment. If you need accommodations around online learning or in classroom accommodations, please be sure to reach out to Student Accessibility Services (SAS) for their assistance (sas@brown.edu, 401-863-9588). Undergraduates in need of short-term academic advice or support can contact an academic dean in the College by emailing college@brown.edu. Graduate students may contact one of the deans in the Graduate School by emailing graduate_school@brown.edu.