CS 120 Fall 2021:
Introduction to Algorithms and their Limitations

Draft Syllabus 8/16/21

Course website:
https://salil.seas.harvard.edu/classes/intro-algorithms-and-their-limitations-fall21

Lecture times: TuTh 9:45-11am Eastern, 114 Western Ave, Allston, Room 2112

Synchronous course preview: Mon 8/23 1-2:15pm Eastern via Zoom (will be recorded). The Zoom link is available from Canvas and will also be posted on the course website. Please email the course staff (below) if you are not able to access it.

Shopping period office hours: Salil will hold virtual advising office hours most days during the course preview period, with a window on Mon 8/23 specifically reserved for CS 120. You can sign up for a 15min slot and find zoom links at http://salil.seas.harvard.edu/.

Limited enrollment lottery: In this first offering, CS 120 will have a limited enrollment of 96 students. If the course is oversubscribed, we will use a random lottery to select the students in the class. *In order to apply, you should add CS 120 to your Crimson Cart and request permission to enroll by Tuesday August 24 at 5pm Eastern Time.* We will inform you about the result of the lottery by Wednesday August 25 at 10am Eastern time.

Course Staff

Instructor: Salil Vadhan (he/him/his; they/them/theirs)
http://salil.seas.harvard.edu/, salil_vadhan@harvard.edu

Head Teaching Fellow: Sanket Shah (he/him/his; they/them/theirs)
sanketshah@g.harvard.edu

Patel Fellow (1:1 tutoring): Helen Cho (she/her/hers)
Reach out through this form, hcho@college.harvard.edu

Teaching Fellows: Sílvia Casacuberta Puig, Ben Dreier, Jennifer Liang, Amanda Fernandez, Melissa Kwan, Edward Pyne, Daniel Rodrigues

Faculty Coordinator: Allison Choat (she/her/hers; they/them/theirs)
achoat@seas.harvard.edu

Whenever possible, please post questions for the course staff on Ed (privately if needed) rather than emailing us, so that we can all see the question and responses.

Overview

Looking at the world around us, we see computers solving problems on incredibly large scales: finding webpages relevant to our internet searches and returning them in sorted order, computing the quickest way to reach a destination given current traffic conditions, matching medical residents to hospitals subject to preferences and capacity constraints. How is this possible? More computing power? Intensive application-specific engineering? While these certainly have had a role to play, in CS 120, you will see and learn how to use general algorithm design principles that cut across application domains and remain relevant even as computing technology changes.

First among these principles is mathematical abstraction, whereby we capture the essence of a computational problem (as well as the notion of a "computer") so that we can develop and analyze solutions independent of an implementation. Given these mathematical abstractions, we can apply a toolkit of basic algorithmic techniques in the search for solutions, and then gain certainty in their correctness and efficiency through rigorous mathematical proofs. Furthermore, the powerful concept of reductions will allow us to identify relationships between computational problems that seem very different on the surface, and thus automatically transfer solutions from one to another.

At the same time, some important computational problems have defied the search for algorithmic solutions. Computer scientists would love to have debugging tools that determine whether their programs can crash, natural scientists would love to have simulators that quickly determine the energy-minimizing states of physical or biological systems, and university registrars would love to be able to automatically schedule classes in a way that optimally maximizes the use of the best classrooms. Why have no scalable algorithms been found for these problems?

In the last part of CS 120, we will see that many important computational problems are inherently unsolvable—they have no general algorithmic solution whatsoever! Others are solvable, but have no efficient algorithm—the computation time inherently grows exponentially with the size of the problem instance. Uncovering these phenomena (known as "uncomputability" and "intractability", respectively) are unique benefits of a mathematically rigorous approach to algorithms. While we may sometimes be satisfied with empirical demonstrations of the performance of an algorithm we have found, a proof seems to be the only way to convince ourselves that there is no algorithm whatsoever.

To summarize, CS 120 aims to give you the power of using mathematical abstraction and rigorous proof to understand computation. Thus equipped, you should be able to design and use algorithms that apply to a wide variety of computational problems, with confidence about their correctness and efficiency, as well as recognize when a problem may have no algorithmic solution. At the same time, we hope you will gain an appreciation for the beautiful mathematical theory of computation that is independent of (indeed, predates) the technology on which it is implemented.

Learning Outcomes

By the end of the course, we hope that you will all have the following skills:

- To mathematically abstract computational problems and models of computation
- To design and implement algorithms using a toolkit of algorithmic techniques
- To recognize and formalize inherent limitations of computation
- To rigorously analyze algorithms and their limitations via mathematical proof
- To appreciate the technology-independent mathematical theory of computation as an intellectual endeavor as well as its relationship with the practice of computing

Prerequisites

Experience with proofs and discrete mathematics at the level of Computer Science 20, and (Python) programming at the level of CS 50. If you haven't taken these courses, we strongly encourage doing Problem Set 0 and turning it in for feedback to gauge your preparation.

Curricular Context

CS 120 is a brand-new introductory course in theoretical computer science, adding to CS 121 (limits & models of computation) and CS 124 (algorithms) in our curriculum. We have added this course in response to growing enrollments in 121 and 124 and to offer a greater variety of introductory theory courses to meet student interests and background. CS 120 is meant to provide a gentler entry into theoretical computer science, which can be taken before CS 121 or CS 124, but these courses will *not* assume CS 120 as a prerequisite. See this FAQ for a more detailed comparison between CS 120, 121, and 124, and the CS Advising site for how CS 120 fits into the old and new CS concentration requirements.

Expectations and Format

This first offering of CS 120 is experimental in several ways, both in the choice of content and in various aspects of the pedagogy. By joining the class, you will be partners in the continued development and improvement of the course. We will be soliciting feedback from you often and expect to be making numerous adjustments based on it **(so everything described below should be seen as tentative)**.

The main planned components of the course are as follows:

Main class meetings (TuTh 9:45-11am): We plan for each class to begin with an approx. 30min active learning exercise done in pairs or groups, building skills and reinforcing content introduced in previous class meetings. These exercises will often require preparation before class, such as studying a proof that you will explain to a partner during the exercise.

The remaining 45min of each class will be an interactive lecture on new material, with occasional in-class poll/survey using the Ed platform (so please do bring a web-enabled device that you can use to answer these questions).

Regular attendance in class is required. The in-class exercises will be developing important skills such as deconstructing and communicating proofs. Additionally, your peers will be depending on your participation and these will be a source of highlights to include in your participation portfolios (described below).

Sections (weekly, times TBD): Sections will be held weekly, at times TBD to accommodate as many students as possible. The sections will consist mainly of guided work on practice problems and interactive review of difficult concepts. You will find sections most useful if you have read and started thinking about the problem sets and the corresponding material in advance. Attendance is optional.

Office hours (every week, at multiple times): The teaching staff will have regular office hours, to help guide you as you work through the material. Before coming to office hours with questions about the problem set (other than clarifications on what is being asked), it is important to invest real effort in trying to solve the problems without staff assistance. If you get stuck, the teaching staff will help you get unstuck, not by giving hints, but by probing your thought process and understanding of the content and/or reviewing material and related problems. The goal is to help you build the skills that will enable you to discover solutions for yourself.

Problem sets: There will be approximately 10 weekly problem sets, typically due Wednesdays at 12 noon. Some of the problems may require deep thought, so you are strongly encouraged to begin them early and to brainstorm in groups of 2-3 students. (See Collaboration Policy below.)

Asynchronous discussions: We will use the *Ed* platform for discussions, Q&A, and announcements.

Participation portfolio: CS 120 is a collective learning experience, where we all should be working to support others' progress. Thus, we expect you to keep a portfolio of the best examples of where your participation helped advance your classmates' learning or helped improve the class. Approximately four times the semester we will ask you to submit the top 3-4 highlights from your portfolio. These can be based on any form of participation: in class/section/office hours, in discussions on Ed, in our feedback surveys, in your collaborations on problem sets, etc.

Exams: There will be an in-class, 75min, closed-book midterm exam on Thursday October 14, and a standard 3-hour final exam scheduled by the Registrar.

Grading

Breakdown. The final exam will comprise 25% of your overall grade in CS 120. The remaining 75% will be split between the problem sets, midterm, and participation portfolios, with equal weight to each problem set or participation portfolio submission or the midterm exam, and the lowest grade dropped. (So if we end up with 11 problem sets and 4 participation portfolio submissions, each one other than the lowest will be worth 75%/(11+4+1-1) = 5% of your grade.)

Grading rubric. The problem sets, participation portfolios, and exams will be graded using the following rubric (with +/- modifiers in case of the final exam):

- N: Not assessable. The assignment is too fragmentary or incomplete for us to assess the level of effort and understanding reached while working on it.
- L: Learning. Significant effort is evident, but there remain important misconceptions or gaps that may limit your ability to apply the skills and/or knowledge in the future (whether in the rest of CS 120, in later CS courses, or outside of your CS education).
- R: Ready to move on. Your work meets all of the learning objectives of this assignment, and contains only minor errors (which we will note in our feedback to you).
- R*: Beyond ready. The work is nearly perfect, goes beyond expectations in its clarity or insight, and/or explores optional extensions.

The reason for this coarse grading scale is to move away from nitpicking over point deductions and focus on the end goal of acquiring sufficient understanding to competently apply what you have learned.

Revisions. The material in CS 120 is largely cumulative, so we wish to see you all reach “R” on all assignments. But mistakes and misunderstandings are a part of the learning process, so we will allow you to transform an L grade to an R grade on a problem set or midterm (not a participation portfolio or final) up to four times during the semester by recording and submitting a three-minute video that clearly explains the misconceptions or gaps in your assignments and the corrected understanding you have developed by studying the solutions. These revisions must be submitted within one week of receiving your graded assignment. We will not accept revisions on assignments that receive an N or R grade.

Letter grades. For a final grade of A, almost all of your assignments and exams should have received R grades (possibly after revision), ideally including some R*’s, and there should be no N grades (except possibly your dropped problem set or midterm). For a final grade of B, the majority of your assignments and exams should have received R grades, again with no N grades (except possibly your dropped problem set or exam). For a final grade of C, you should have a mix of R and L grades, with very few N’s.

Late days. You have a total of six late days that you can use for problem sets or participation portfolios, using at most three on any one assignment. (That is, an assignment beyond 3 days late will automatically receive an N grade.) Do not needlessly use up your late days at the beginning of the semester; they are meant for accommodating unforeseen circumstances or crunches from your other classes or other aspects of your life. Any extensions beyond these late days require a note from your resident dean.

Diversity and Inclusion[1]

We would like to create a learning environment in our class that supports a diversity of thoughts, perspectives and experiences, and honors your identities (including race, gender, class, sexuality, socioeconomic status, religion, ability, etc.). We (like many people) are still in the process of learning about diverse perspectives and identities. If something was said in class (by anyone) that made you feel uncomfortable, please talk to us about it. If you feel like your performance in the class is being impacted by your experiences outside of class, please don't hesitate to come and talk with us. As a participant in course discussions, you should also strive to be open-minded and respectful of your classmates. (Statement)

Health Accommodations[2]

If you have a physical or mental health condition that affects your learning or classroom experience, please let us know as soon as possible so that we can do our best to support your learning (at minimum, providing all of the accommodations listed in your AEO letter if you have one). (Statement adapted from one by Prof. Krzysztof Gajos.)

Support Structures[3]

Everyone can benefit from support during challenging times. If you experience significant stress or worry, changes in mood, or problems eating or sleeping this semester, whether because of CS 120 or other courses or factors, please do not hesitate to reach out to Salil or other members of the course staff. Not only are we happy to listen and discuss how we can help you cope in CS 120, we can also refer you to additional support structures on campus, including, but not limited to, the below.

- Bureau of Study Counsel
- InTouch
- Counseling and Mental Health Services, 617-495-2042
- Let's Talk
- Room 13, 617-495-4969

---

[1] Based on text by Dr. Monica Linden at Brown University.
[2] Based on text by Prof. Krzysztof Gajos at Harvard University.
[3] Based on text in the Harvard CS50 Syllabus.

Collaboration Policy

Students are encouraged to discuss the course material and the homework problems with each other in small groups (2-3 people). Discussion of homework problems may include brainstorming and talking through possible solutions, but should not include one person telling the others how to solve the problem. In addition, each person must write up their solutions independently, and these write-ups should not be checked against each other or passed around. While working on your problem sets, you should not refer to existing solutions, whether from other students, past offerings of this course, materials available on the internet, or elsewhere. All sources of ideas, including the names of any collaborators, must be listed on your submitted homework along with a brief description of how they influenced your work.

In general, we expect all students to abide by the Harvard College Honor Code. We view us all (teaching staff and students) as engaged in a *shared mission* of learning and discovery, not an adversarial process. The assignments we give and the rules we set for them (such as the collaboration policy) are designed with the aim of maximizing what you take away from the course. We trust that you will follow these rules, as doing so will maximize your own learning (and thus performance on exams) and will maintain a positive educational environment for everyone in the class. We welcome and will solicit feedback from you about what more we can do to support your learning.

Content

There is no required textbook for CS 120. We will be drawing on material from a number of different sources; pointers to or extracts from relevant texts will be provided as we proceed. A tentative content breakdown will be available and continuously updated on the course website.