Intro to Algorithms & their Limitations

Salil Vadhan '95 (he/they)

Head TF: Sanket Shah

Announcements

Turn on video

- Start recording
- If Zoom problems, check course website at salil.seas.harvard.edu
- Request to enroll by Tomorrow 5pm ET (random lottery)
- My OH today 2:30–4pm + Wed 11:30–12 + ? (sign up on my website)
- Handouts: drafts of ps 0, syllabus, content plan, textbooks

Agenda

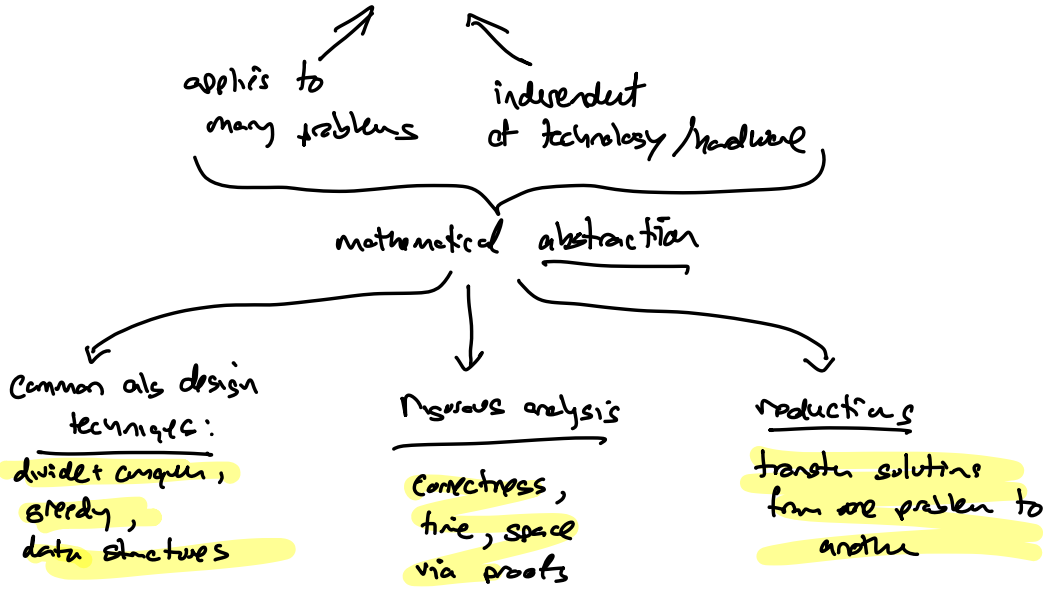
- Overview of learning objectives + content
- Course operation
- Q & A

Motivation

Algorithms solving problems on a massive scale, e.g. web search, Google maps, online dating

How? faster hardware? application-specific engineering?

CS120: efficient, general-purpose algorithm design



Algorithmic Problem Areas We'll Study

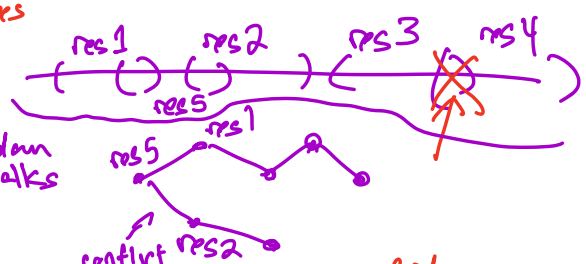
→ Storing, sorting, searching datasets (~6 classes)

- ordered vs. unordered data
- static vs. dynamic data → balanced binary search trees
- deterministic vs. randomized algs

data structure

ordered : interval scheduling

collection of time window reservations



→ Graph algorithms (~4 classes)

- graph search – BFS, DFS, Random walks
- graph coloring + independent sets
- matching – "independent" set of edges

find a edge-free (conflict-free) set of nodes as large as possible

→ Geometric algorithms (2 classes if time permits)

- apps: graphics, robotics, ML

→ Logic algorithms $(x_1 \vee \neg x_2 \vee x_5) \wedge (x_3 \vee x_7) \wedge \cdots$

- Propositional satisfiability (SAT/MIP/SMT solvers)
- apps: program verification

2-coloring: partition nodes into two edge-free sets

2-coloring easy, 3-coloring hard

Limits of Algorithms

Why don't we have:

- perfect software debuggers? → uncomputability: provably no general algorithm
- fast algorithms for combinatorial scheduling problems?
- fast algorithms for predicting behavior of natural systems?

→ intractability: provably no polynomial-time algorithm (assuming $P \neq NP$)

(Extended) Church-Turing Thesis: these are technology-independent limitations

Learning Outcomes

- mathematical abstraction
- design + implement algorithms
- recognize + formalize limits
- rigorously analyze
- appreciate the theory

Course Question ⟶ Syllabus