# CS4102 Algorithms

Spring 2020

# Today's Keywords

- Course feedback
- Differential Privacy
- (NP-Completeness)
- Final words!

Collaborate policies

# CLRS Readings

NONE

# Homeworks

- HW9 due Tues. April 28 at 11pm
  - Reductions, Graphs
  - Written (LaTeX)
- HW10C due tonight at same time
- Note: no late submissions after Weds., April 29, 5pm

# Final Exam

- Take-home exam
  - Open book, open notes, open course material **only**
  - NO collaboration, NO internet (i.e. no "googling")
    - See full rules etc at https://bit.ly/cs4102-final-s20
  - It will be on Collab under "Tests and Quizzes"
    - Variety of question formats
    - You may want to write pseudocode in a text editor and copy/paste into Collab
  - Must take between 6pm Thursday (4/30) and 1pm Sunday (5/3)
    - 2 hours to submit once started

7 - 10 pm
Sat. night

# Final Exam Rules

- From that web-page on what's allowed:
  - OK: Anything created by course staff (on the Web), book, notes, your HWs, Piazza answers
  - NOT allowed: Viewing any other site or resource on the Internet; communication or assistance with anyone during exam; communication after that might help someone else
- Note "blackout" on Piazza answers about exam topics

# Final Exam

~~75 minutes~~

120 minutes

- Review
  - Practice Exam out by tomorrow
    - We are releasing last semester's final, which will give some new problems but our final will be different!
  - Practice Exam solutions out on Wednesday
  - Review sessions Wednesday, 5pm (Hott), and Thursday (Horton)
- SDAC: please send us an email so that we can coordinate
  - Contact us by Wed., April 29, 5pm EDT

other

# Follow-up and Feedback on Some Things

- In late February, someone from UVA's CTE observed Horton's class and talked with students
- Horton discussed results with CTE person
- A few changes made before spring break, and then… ☹
- But let's discuss what I learned

# What you told CTE

- What impedes learning?
  - Moves through material too quickly. Slides are confusing. Too long to get graded HW. Poor feedback on coding HW.
- Suggestions for instructor
  - Use pen to interact with slides or whiteboard. Provide opportunities to work on problems in class. Make additional resources available other than slides or textbook. Go slower through slides. Periodically check for student understanding.
- Attendance: I was curious. Thanks for answers!

- Youtube videos
- worksheets
- Common practice problems
- code?

# Horton's Responses

- Moved to using pen with PPT slides before spring break (and since)
- In-class exercises, activities, checks
  - First one: matrix chaining bottom-up DP algorithm
  - Had others planned for after spring break
  - Questions for you:  takes time, worth it?

# More Responses, and Your Feedback

- About slides, pace: some observations on current course design
  - Slides, example problems, proofs, implementation details
  - Relationship with style of HW problems (problem solving, proof, coding)

# One More Issue: Online Situation

- (We will talk about collaboration policy later, but comments welcome now.)
  - when O/H scheduled
- What worked well in how we handled online?
- What could have worked better?
  - Piazza, Discord for TA O/H, Zoom for Instructor O/H
  - Pre-recorded videos, then class time for Q&A

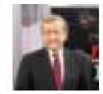
how far later, next day

Activities in Q&A practice


President Trump Expected to Shrink Bears Ears by as Much as 90 Percent


Ministers Look to Revive Martin Luther King's 1968 Poverty Campaign


Alabama's Disdain for Democrats Looms Over Its Senate Race


ABC Suspends Reporter Brian Ross Over Erroneous Report About Trump

# *As Computer Coding Classes Swell, So Does Cheating*

By JESS BIDGOOD and JEREMY B. MERRILL MAY 29, 2017

A Computer Science 50 course at Harvard in 2013. Last fall, more than 60 computer science students were referred to the university's honor council, which investigates cheating allegations. Joseph Ong

The New York Times



# *As Computer Coding Classes Swell, So Does Cheating*



*BIZ & IT —*

# Code copypasta increasingly common in CS education

Roughly 22 percent of Stanford honor code violations involve plagiarism in ...

RYAN PAUL - 2/12/2010, 5:11 PM

A Computer Science 50 course at Harvard in 2013. Last fall, more than 60 computer science students were referred to the university's honor council, which investigates cheating allegations. Joseph Ong

SECTIONS HOME SEARCH

The New York Times

President Trump Expected to Shrink Bears Ears by as Much as 90 Percent

Ministers Look to Revive Martin Luther King's 1968 Poverty Campaign

Alabama's Disdain for Democrats Looms Over Its Senate Race

ABC Suspends Reporter Brian Ross Over Erroneous Report About Trump

## *As Computer Coding Classes Swell, So Does Cheating*

ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE FORUMS SIGN IN

*BIZ & IT —*

# Code copypasta increasingly common in CS education

Roughly 22 percent of Stanford honor code violations involve plagiarism in ...

RYAN PAUL - 2/12/2010, 5:11 PM

# THE DAILY ILLINI

The independent student newspaper at the University of Illinois

NEWS SPORTS OPINIONS LIFE & CULTURE SPECIAL SECTIONS LONGFORM BUZZ CLASSIFIEDS

## College of Engineering piloting program to combat cheating

**Top Stories**

# What are some reasons students cheat?

over-collaboration

helpful, better than other classes

proofs

finding a group might be hard

find during office hours

common solution

mix, begin??

# Our In-Person Academic Integrity Policy

work?

You are encouraged to collaborate with up to 4 other students, but all work submitted must be your own independently written solution. List the computing ids of all of your collaborators in the comments at the top of each submitted file. Do not share written notes, documents (including Google docs, Overleaf docs, discussion notes, PDFs), or code. Do not seek published or online solutions, including pseudocode, for this assignment. If you use any published or online resources (which may not include solutions) when completing this assignment, be sure to cite them. Do not submit a solution that you are unable to explain orally to a member of the course staff. Any solutions that share similar text/code will be considered in breach of this policy. Please refer to the syllabus for a complete description of the collaboration policy.

# Our Online Academic Integrity Policy

You are encouraged to collaborate with up to 4 other students, but all work submitted must be your own *independently* written solution. List the computing ids of all of your collaborators in the comments at the top of the tex file. You **are** permitted to collaborate through online tools such as Google Docs, interactive whiteboards, Google Meet, Google Hangouts, Zoom, Skype, etc, however you **must** limit written/typed details to high-level algorithm design. Each person is responsible for taking those ideas and turning them into pesudocode and a writeup. Do **NOT** copy and paste from shared documents, which includes re-typing verbatim or trying to disguise text that you are essentially copying. Over-collaboration of that form is fairly easy to detect with plagiarism tools. **Do not seek published or online solutions, including pseudocode, for this assignment.** If you use any published or online resources (which may not include solutions) when completing this assignment, be sure to cite them. Do not submit a solution that you are unable to explain orally to a member of the course staff. Any solutions that share similar text/code will be considered in breach of this policy. Please refer to the syllabus for a complete description of the collaboration policy.

# Our Collaboration Policy and Transition

transitin – good much didn't change

- What worked in our transition to an online course?
- What could we have improved upon?
- What did other instructors do that you found helpful?
- What is a ”good” collaboration policy?
- How do collaboration policies and academic integrity policies change when transitioning online?

# Our Collaboration Policy and Transition

# Our Collaboration Policy and Transition

# Differential Privacy

- Gives a way to probabilistically answer questions about data without giving away its content
- You can get statistical certainty on the answer
- We're going to use a simple example

preserve privacy

# Scheme

- Flip a coin:
  - If Heads, respond “yes”
  - If Tails, truthfully answer an embarrassing question:
- Questions
  - Do Nate and I share a minecraft server?
  - Have you ever blacked out?
  - Are you a virgin?
  - Can you give us the answers to the final ahead of time?

# Scheme

- Flip a coin:
  - If Heads, respond “yes”
  - If Tails, truthfully answer an embarrassing question:
    - Have I ever tried to impress a girl with algo and failed epically?
    - Have I ever streaked the lawn?
    - Have I ever drank before class?
    - Have I ever cheated
    - Is the 11am section better than the 2pm?

    - Do I find any of my coworkers attractive?
    - Do I have any tats or piercings?
    - Have I ever had an awkward date?
    - Do I drive a red punch buggy?
    - Have I ever pooped myself as a teenage+?
    - Do I think I’m smart enough to have something named after me?
    - Was UVA my second/ worst choice to work at?

# Scheme

- Flip a coin:
  - If Heads, respond “yes”
  - If Tails, truthfully answer an embarrassing question:

Have I ever been mistaken for a student?
Have I ever been drinking at the corner and came upon a student?
Have I ever had an encounter with the fuzz
Would I like a soup or a salad?
Have I ever used bubblesort?
Have I ever actually used bogosort?
Do I discuss algorithms on dates?
Am I on Tinder/bumble?
Have I ever used a CS pickup line?
Do I compare myself to Mark Floryan?
Is there a better programmer in the CS department?
Disp

# Scheme – Embarrassing Questions

- Flip a coin:
  - If Heads, respond "yes"
  - If Tails, truthfully answer an embarrassing question
- Questions

- have I peed on myself
- have I cheated on an assignment
-

heads → yes

tails NO

# How does it work

- Assume everyone follows the rules
- We know 50% of the answers were "yes" because the coin landed **heads**
  - Reduce count of "yes" responses by 50% of total responses
  - Calculate "true yes" using this reduced "yes" count and the updated total (half of original total)
- Example: We get 40 yes and 20 no answers.
  - Expect 50% of total answers are "yes" due to coin-flip
  - Eliminate 50% or 30 "yes" answers
  - Leaves 10 "yes" and 20 "no" responses, total of 30. Estimate of "true yes" is 33%.

**Is your privacy protected?**

- Consider a person who answers "no"
  - We know this person is a "true no". (Is this OK? Probably.)
- Consider a person who answers "yes"
  - Most people (≥ 50%) who answered "yes" only did so because the coin landed **heads**
  - It's still **more** likely that this person **is a "true no"**
  - So if we learn you said "yes", then we can't justifiably accuse you being a "true yes"

# An Example – Your Turn!

- Flip a coin: (on Google, search “flip a coin”)
  - If Heads, respond “yes”
  - If Tails, truthfully answer an embarrassing question

**Have you ever streaked the lawn?**

# Impagliazzo’s 5 Worlds

Describes what computer science might look like depending on how certain open questions are answered.

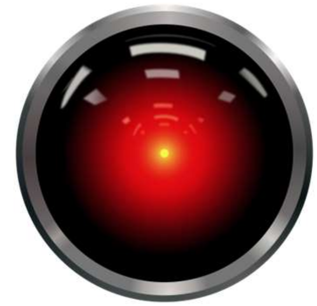
- Algorithmica
- Heuristica
- Pessiland
- Minicrypt
- Cryptomania

# Gauss vs. Büttner

Büttner's goal: embarrass Gauss

- Come up with a problem which Gauss finds difficult but Büttner can solve quickly
  1. Come up with a graph and a Vertex Cover together
  2. Give the graph to Gauss
  3. When Gauss is stumped show the Vertex Cover

# Algorithmica



$P = NP$

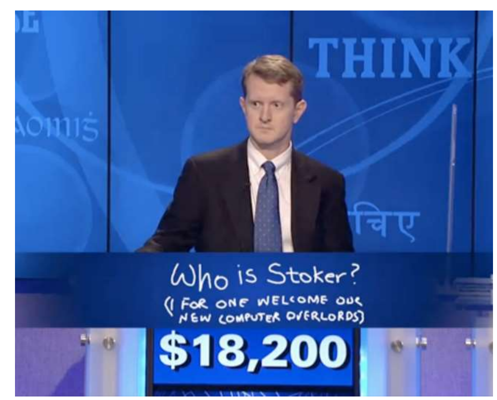
- NP problems solvable efficiently
- Gauss can quickly find the solution to Büttner’s problem
- Gauss is not embarrassed

Advantages:

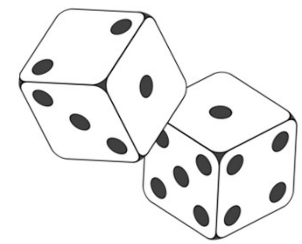
- VLSI Design
- Strong AI
- Cure for cancer?

Disadvantages:

- No privacy
- Computers take over



# Heuristica



$P \neq NP$ in worst case, $P = NP$ on average

- Time to come up with a problem ≈ time to solve it
- Büttner can give hard problems, but it's hard to find them
- Gauss is not embarrassed

Advantages:

- Maybe similar to Algorithmica
- Depends on real-world distributions

Disadvantages:

- Bad real world distributions could make things hard to solve

# Pessiland

$P \neq NP$ on average, one-way functions don't exist

- Hard problems easy to find, but *solved* hard problems difficult to find
- Gauss can be stumped, but Büttner does no better

Advantages:

- Universal Compression
- Reverse Engineering
- Derandomization

Disadvantages:

- No crypto
- No algorithmic advantages
- Progress is slow

# Minicrypt

One-way functions exist, no public key cryptography

- Büttner can give hard problems to Gauss and also know their solutions
- Gauss is embarrassed

Advantages:

- Private key crypto
- Can prove identity

Disadvantages:

- No electronic currencies

# Cryptomania

Public Key Crypto Exists

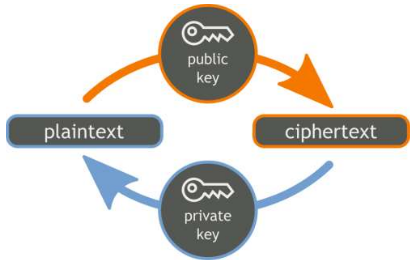
- Büttner can come up with problems and solutions, then share the solution with all other students
- Gauss is very embarrassed

Advantages:

- Secure computation
- Signatures
- Bitcoin, etc.

Disadvantages:

- Algorithmic progress will be slow



# Does P=NP?

| | $P \neq NP$ | $P = NP$ | Ind | DC | DK | DK and DC | other |
|---|---|---|---|---|---|---|---|
| 2002 | 61(61%) | 9(9%) | 4(4%) | 1(1%) | 22(22%) | 0(0%) | 3(3%) |
| 2012 | 126 (83%) | 12 (9%) | 5 (3%) | 5 (3%) | 1(0.6%) | 1 (0.6%) | 1 (0.6%) |

# When Will P=NP be resolved?

| | 02–09 | 10–19 | 20–29 | 30–39 | 40–49 | 50–59 | 60–69 | 70–79 |
|---|---|---|---|---|---|---|---|---|
| 2002 | 5(5%) | 12(12%) | 13(13%) | 10(10%) | 5(5%) | 12 (12%) | 4(4%) | 0(0%) |
| 2012 | 0(0%) | 2(.01%) | 17(11%) | 18(12%) | 5(3%) | 10 (6.5%) | 10 (6.5%) | 9(6%) |

| | 80–89 | 90–99 | 100–109 | 110–119 | 150–159 | 2200–3000 | 4000–4100 |
|---|---|---|---|---|---|---|---|
| 2002 | 1(1%) | 0(0%) | 0(0%) | 0(0%) | 0(0%) | 5(5%) | 0(0%) |
| 2012 | 4(3%) | 5(3%) | 2(1.2%) | 5(3%) | 2(1.2%) | 3(2%) | 3(2%) |

| | Long Time | Never | Don't Know | Sooner than 2100 | Later than 2100 |
|---|---|---|---|---|---|
| 2002 | 0(0%) | 5(5%) | 21(21%) | 62(62%) | 17 (17%) |
| 2012 | 22(14%) | 5(3%) | 8(5%) | 81(53%) | 63 (41%) |

# Notable Statements on P vs NP

**Scott Aaronson** I believe P $\neq$ NP on basically the same grounds that I think I won't be devoured tomorrow by a 500-foot-tall robotic marmoset from Venus, despite my lack of proof in both cases.

Suggested rephrased question:

*will humans manage to prove* P $\neq$ NP *before they either kill themselves out or are transcended by superintelligent cyborgs? And if the latter, will the cyborgs be able to prove* P $\neq$ NP*?*

**Neil Immerman** P $\neq$ NP will be resolved somewhere between 2017 and 2034, using some combination of logic, algebra, and combinatorics.

**Donald Knuth:** (Retired from Stanford) It will be solved by either 2048 or 4096. I am currently somewhat pessimistic. The outcome will be the truly worst case scenario: namely that someone will prove "P=NP because there are only finitely many obstructions to the opposite hypothesis"; hence there will exists a polynomial time solution to SAT but we will never know its complexity!

# Let's Reflect

# Course Objectives

*Students who complete the course will:*

1. Comprehend **fundamental ideas in algorithm analysis**, including: time and space complexity; identifying and counting basic operations; order classes and asymptotic growth; lower bounds; optimal algorithms.
2. Apply these fundamental ideas to **analyze and evaluate important problems and algorithms in computing**, including search, sorting, graph problems, and optimization problems.
3. **Apply appropriate mathematical techniques in evaluation and analysis**, including limits, logarithms, exponents, summations, recurrence relations, lower-bounds proofs and other proofs.
4. **Comprehend, apply and evaluate the use of algorithm design techniques** such as divide and conquer, the greedy approach, dynamic programming, and exhaustive or brute-force solutions.
5. Comprehend the fundamental ideas related to **the problem classes NP and NP-complete**, including their definitions, their theoretical implications, Cook's theorem, etc. Be exposed to the design of polynomial reductions used to prove membership in NP-complete.

# Your New Knowledge and Skills

- Able to compare algorithms' efficiency
  - Without coding them, without running on data
  - Able to talk about efficiency formally, mathematically
- Problem solving skills for a variety of challenging problems
- Algorithm design strategies: divide and conquer, greedy, dynamic programming, brute-force
- Ability to argue about an algorithm's correctness or that it's optimal
  - Proof techniques. Exchange argument, cut theorem, decision trees, etc.
  - Lower bounds proofs to show optimality or minimum work required

# Your New Knowledge and Skills (2)

- How solve certain types of problems
  - "Search" problems; optimization problems; graph problems; etc.
- Theoretical CS issues regarding hard problems
  - You know a set of intractable problems
  - Knowledge of the existence of NP-Hard problems, and importance of NP-C problems
  - Understand the practical and theoretical implications of P=NP question

# Last Words!

- The topics of CS4102 combine:
  - practical problem solving,
  - understanding of data structures,
  - useful mathematical analysis of efficiency,
  - proof techniques for important properties of algorithms,
  - a tour of problems and design strategies,
  - theory about the nature of problems that has important implications about the world.
- Congratulations for completing your journey through this amazing CS course!