

Algorithms & Models of Computation CS/ECE 374, Fall 2017

# Administrivia, Introduction

Lecture 1 Tuesday, August 29, 2017

Sariel Har-Peled (UIUC

# Instructional Staff Instructor: Sariel Har-Peled 10<sup>9</sup> students. 9 Teaching Assistants 16 Undergraduate Course Assistants Office hours: See course webpage Contacting us: Use *private notes* on Piazza to reach course staff. Direct email only for sensitive or confidential information.

Fall 2017

## Online resources

Webpage: General information, announce course policies	
http://courses.engr.illinois.edu/cs	s374/fa2017/
Gradescope: Homework submission and g requests	rading, regrade
Moodle: Quizzes, solutions to homeworks	s, grades
Piazza: Announcements, online questions contacting course staff (via private notes)	
See course webpage for links	
Important: check Piazza at least once each da	ıy.
Sariel Har-Peled (UIUC) CS374 5	Fall 2017 5 /

# Grading Policy: Overview **Quizzes:** 0% for self-study A Homeworks: 28% **Midterm exams:** 42% (2 × 21%) Final exam: 30% (covers the full course content) Midterm exam dates: Midterm 1: Monday October 2, 7-9pm. 2 Midterm 2: Monday November 13: 7-9pm. No conflict exam offered unless you have a valid excuse. Fall 2017

#### Preregs and Resources

- Prerequisites: CS 173 (discrete math), CS 225 (data structures)
- Recommended books: (not required)
  - Introduction to Theory of Computation by Sipser
  - Introduction to Automata, Languages and Computation by Hopcroft, Motwani, Ullman
  - 3 Algorithms by Dasgupta, Papadimitriou & Vazirani. Available online for free!
  - Algorithm Design by Kleinberg & Tardos
- Lecture notes/slides/pointers: available on course web-page

#### Additional References

- Lecture notes of Jeff Erickson, Sariel Har-Peled, Mahesh Viswanathan and others
- 2 Introduction to Algorithms: Cormen, Leiserson, Rivest, Stein.
- S Computers and Intractability: Garey and Johnson.

# Homeworks

Sariel Har-Peled

- Self-study guizzes each week on *Moodle*. No credit but strongly recommended.
- One homework every week: Due on Wednesdays at 10am on Gradescope. Assigned at least a week in advance.
- O Homeworks can be worked on in groups of up to 3 and each group submits one written solution (except Homework 0).
- Important: academic integrity policies. See course web page.

Fall 2017

# More on Homeworks

- No extensions or late homeworks accepted.
- To compensate, nine problems will be dropped. Homeworks typically have three problems each.
- Important: Read homework FAQ/instructions on website.

Sariel Har-Peled (UIUC)	CS374	9	Fall 2017	9 / 26

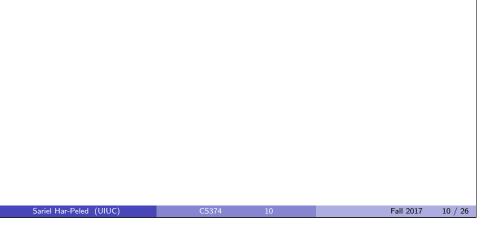
# Advice

- 4ttend lectures, please ask plenty of questions.
- 2 Attend discussion sessions.
- On't skip homework and don't copy homework solutions. Each of you should think about *all* the problems on the home work do not divide and conquer.
- Use pen and paper since that is what you will do in exams which count for 75% of the grade. Keep a note book.
- Study regularly and keep up with the course.
- This is a course on problem solving. Solve as many as you can! Books/notes have plenty.
- This is also a course on providing rigorous proofs of correctness. Refresh your 173 background on proofs.
- **③** Ask for help promptly. Make use of office hours/Piazza.

Fall 2017 11 / 26

## Discussion Sessions/Labs

- **1** 50min problem solving session led by TAs
- I wo times a week
- **③** Go to your assigned discussion section
- Bring pen and paper!



# Homework 0

- HW 0 is posted on the class website. Quiz 0 available on Moodle.
- W 0 due Wednesday, September 6, 2017 at 10am on Gradescope.
- Groups of size up to 3.

374 1

## Miscellaneous

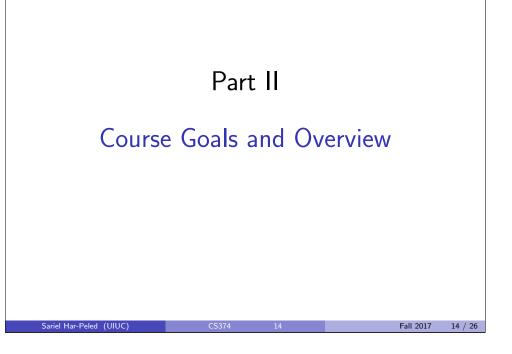
Please contact instructors if you need special accommodations.

Lectures are being taped. See course webpage.

Sariel Har-Peled (UIUC)	CS374	13	Fall 2017	13 / 26

# High-Level Questions

- Modeling: States/Graphs/Recursion/Algorithms.
- Algorithms
  - What is an algorithm?
  - **2** What is an *efficient* algorithm?
  - **③** Some fundamental algorithms for basic problems
  - Broadly applicable techniques in algorithm design
- **③** What is a mathematical definition of a computer?
  - **1** Is there a formal definition?
  - Is there a "universal" computer?
- What can computers compute?
  - Are there tasks that our computers cannot do?



# Course Structure

Course divided into three parts:

- Basic automata theory: finite state machines, regular languages, hint of context free languages/grammars, Turing Machines
- Algorithms and algorithm design techniques
- Undecidability and NP-Completeness, reductions to prove intractability of problems

CS374

#### Goals

#### 1

- 2 Learn/remember some basic tricks, algorithms, problems, ideas
- Output Understand/appreciate limits of computation (intractability)
- Appreciate the importance of algorithms in computer science and beyond (engineering, mathematics, natural sciences, social sciences, ...)

Sariel Har-Peled (UIUC)	CS374	17	Fall	2017 17 / 26

# Models of Computation vs Computers

- Model of Computation: an "idealized mathematical construct" that describes the primitive instructions and other details
- Computer: an actual "physical device" that implements a very specific model of computation

#### Models and devices:

- Algorithms: usually at a high level in a model
- **2** Device construction: usually at a low level
- Intermediaries: compilers
- How precise? Depends on the problem!
- 9 Physics helps implement a model of computer
- O Physics also inspires models of computation

# Historical motivation for computing

- Fast (and automated) *numerical calculations*
- Q Automating mathematical theorem proving

Sariel Har-Peled

Adding Numbers	
Problem Given two $n$ -digit numbers $x$ and $y$ , compute their sum.	
Basic addition	h
3141 <u>+7798</u> 10939	
Sariel Har-Peled (UIUC)         CS374         20         Fail 2017         20 /	26

CS3-

Fall 2017

# Adding Numbers

c = 0for i = 1 to n do  $z = x_i + y_i$ z = z + cIf (z > 10)c = 1z = z - 10 (equivalently the last digit of z) Else c = 0print zEnd For If (c == 1) print c

- Primitive instruction is addition of two digits
- **2** Algorithm requires O(n) primitive instructions

# Time analysis of grade school multiplication

- Each partial product:  $\Theta(n)$  time
- **2** Number of partial products:  $\leq n$
- Adding partial products: n additions each  $\Theta(n)$  (Why?)
- Total time:  $\Theta(n^2)$

Sariel Har-Peled (UIUC)

Is there a faster way?

# Multiplying Numbers

Problem Given two *n*-digit numbers *x* and *y*, compute their product.

#### Grade School Multiplication

Compute "partial product" by multiplying each digit of y with x and adding the partial products.

	$\frac{\times 2}{25}$	2		
Sariel Har-Peled (UIUC)	CS374	22	Fall 2017	22 / 26

# Fast Multiplication

Best known algorithm:  $O(n \log n \cdot 2^{O(\log^* n)})$  time [Furer 2008]

Previous best time: O(n log n log log n) [Schonhage-Strassen 1971]

**Conjecture:** there exists an  $O(n \log n)$  time algorithm

We don't fully understand multiplication! Computation and algorithm design is non-trivial!

Fall 2017

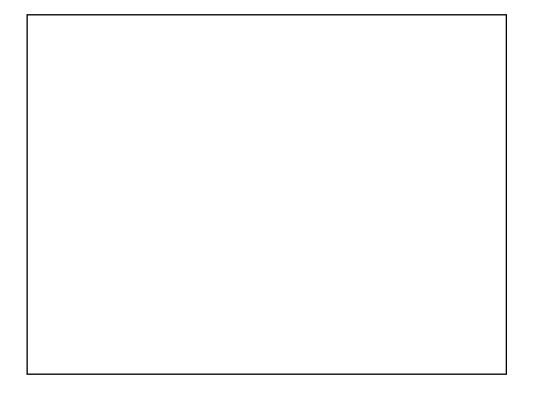
# Post Correspondence Problem

Given: Dominoes, each with a top-word and a bottom-word.



Can one arrange them, using any number of copies of each type, so that the top and bottom strings are equal?

	abb	ba	abb	а	abb	b		
	а	bbb	а	ab	baa	bbb		
Sariel Har-Peled	(UIUC)		CS374	25			Fall 2017	25



# Halting Problem

**Debugging problem:** Given a program M and string x, does M halt when started on input x?

**Simpler problem:** Given a program *M*, does *M* halt when it is started? Equivalently, will it print "Hello World"?

One can prove that there is no algorithm for the above two problems!



CS374

26

Fall 2017