January 17, 2022 5:28 PM

CS/ECE 374 A : Intro to Algorithms & Models of Computation

http://courses.engr.illinois.edu/cs374/sp2022/A

Instructors: Timothy Chan, Ruta Mehta

Lectures & Labs:

first week: completely online

afterwards: lectures in-person (ECEB 1002) & recorded
& streamed live on zoom

(lecture scribbles will be on web page)

labs: mix of in-person & zoom sections (TBA)

Office Hrs: online (zoom / Discord) (more details later)

Piazza

(note: please be courteous & respectful to others!)

HWs:

11 Guided Problem Sets ("GPS") on PrairieLearn (autograded)

weight = 1 HW problem

\+ 11 Written Homeworks

each = 2 HW problems

may work in groups ≤ 3

⇒ total = 33 HW problems

no late HWs!

(but may drop 6 problems)

(if illness / extenuating circumstances, ask instructors ...)

Exams:

Midterm 1: Feb 21 Mon 7p-9:30p (conflict: TBA)

Midterm 2: Apr 11 Mon 7p-9:30p

Final: TBA

(proctor via zoom)

Grades:

HWs	28%
Midterm 1	21%
Midterm 2	21%
Final	30%

option 1: fixed cut-offs

option 2: curved

take better of two

(see web pages)

Overview

introduction to CS theory

Goal 1: how to solve problems (efficiently)

algorithm design & analysis

Goal 2: how to show that a problem can't be solved (efficiently)

mathematically prove

Outline

Part I. Models of Computation

→ finite automata ↔ regular exprs

context-free grammars

Turing machines

Part II. Algorithm Design Techniques

divide & conquer

dynamic programming*

greedy

graph algorithms

Part III. NP-Completeness* & Undecidability

Ex1 Given n numbers,

(3SUM) do there exist 3 numbers summing to 100?

e.g. 81, 95, 43, 20, 32, 74, 25

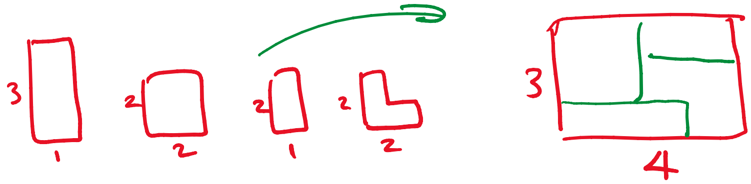
brute-force alg'm: $O(n^3)$ time

smarter alg'm: $O(n^2)$ time ...

fastest? OPEN

$\sim O\left(\frac{n^2}{\log^2 n}\right)$ [c' 2018]

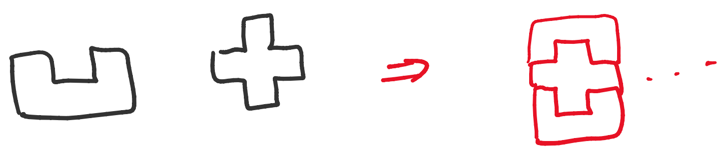
Ex2 Given n polygons & 1 rectangle,
can they be packed in rectangle?



no efficient alg'm believed to be possible

↑ NP-complete

Ex3 Given n polygons,
Can they tile the entire plane?
(assuming infinite copies)



no alg'm possible (undecidable)

Part I. Models of Computation

Math Preliminaries

Strings

A string is a finite sequence of symbols from a finite set Σ ← alphabet

inputs to program or program etc.

e.g. strings over $\Sigma = \{0,1\}$

1001 01 101 0

Let ε denote the empty string.

Let Σ^* denote {all strings over Σ}.

Let Σ denote ...

Let Σ^* denote $\{$ all strings over $\Sigma \}$.

Let x, y be strings.

a) length $|x|$

e.g. $|1001| = 4$, $|\varepsilon| = 0$

b) concatenation xy

e.g. $x = 01$, $y = 1001 \Rightarrow xy = 011001$

$$(xy)z = x(yz) \qquad xy \neq yx$$

$$|xy| = |x| + |y|$$

$$\varepsilon x = x, \qquad x\varepsilon = x$$

c) ith power $x^i = \underbrace{xx\ldots x}_{i \text{ times}}$

e.g. $(101)^3 = 101101101$

$$\begin{cases} x^0 = \varepsilon \\ x^i = x \cdot x^{i-1} \end{cases} \qquad |x^i| = i|x|$$

d) x is a substring of y if

$y = wxz$ for some strings w, z

(prefix if $w = \varepsilon$, suffix if $z = \varepsilon$)

e) other ops: $x^R =$ reverse of x

$$\left(x^R = \begin{cases} \varepsilon & \text{if } x = \varepsilon \\ y^R a & \text{if } x = ay \text{ for some } a \in \Sigma, y \in \Sigma^* \end{cases} \right)$$

[convention: symbols $a, b, c, \ldots$ strings $x, y, z, \ldots$]

$$(xy)^R = y^R x^R \qquad \text{(lab 1a)}$$

Languages

A language is a set L of strings (over Σ)
(i.e. $L \subseteq \Sigma^*$)

e.g. $\{1001, 01, 101, 0\}$

finite, boring! $\{$ all words in English dictionary $\}$ over $\Sigma = \{\text{'a'}, \ldots, \text{'z'}\}$

infinite, more interesting $\{x \in \{0,1\}^* : |x| \text{ is odd}\}$

$\{$ all prime numbers written in binary $\}$

$\{$ all syntatically valid python programs $\}$

(languages can encode all decision problems)

Let L_1, L_2 be languages.

a) union $L_1 \cup L_2$
intersection $L_1 \cap L_2$
complement $\overline{L_1} = L_1^c = \Sigma^* \setminus L_1$
difference $L_1 \setminus L_2 = L_1 \cap \overline{L_2}$

b) concatenation
$L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$

e.g. $L_1 = \{0, 00\}$, $L_2 = \{1, 01\}$
$L_1 L_2 = \{01, 0001, 001\}$

e.g. $L_1 = \{0, 00, 000, \ldots\} = \{0^i : i \geq 1\}$
$L_2 = \{1, 11, 111, \ldots\} = \{1^i : i \geq 1\}$
$L_1 L_2 = \{ \cancel{0^i 1^i : i \geq 1} \}$ wrong
$L_1 L_2 = \{0^i 1^j : i \geq 1, j \geq 1\}$

c) ith power: $L^i = LL \cdots L$

c) i^{th} power: $L^i = \underbrace{LL\cdots L}_{i \text{ times}}$

e.g. $\{1, 01\}^2 = \{11, 0101, 101, 011\}$

$L^0 = \{\varepsilon\}$

$L^i = L \cdot L^{i-1}$

d) <u>Kleene</u> <u>star</u>

$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup \cdots$$

e.g. $\{01\}^* = \{\varepsilon, 01, 0101, 010101, \ldots\}$

$\{1, 01\}^* = \{\varepsilon, 1, 01,$
$11, 0101, 101, 011,$
$111, 010101, 10101, 0111, \ldots, \ldots\}$
$1011, \ldots$

$= \{ x \in \{0,1\}^* :$
x does not contain 00 as a substring
→ & does not end 0 $\}$

(Proof?)

$\{0,1\}^* =$ all strings over $\{0,1\}$