Overview

theory course on ① problem solving
in CS, ~~writing a program~~ designing efficient algorithms

② or demonstrating that a problem can't be solved efficiently
mathematically proving

Outline

I. Models of Computation
finite automata ↔ regular exprs
context-free grammars
Turing machines

II. Algorithm Design
divide & conquer
dynamic programming
greedy
graph algorithms

III. Undecidability & NP-Completeness

Ex1 Given n numbers,
are there 3 numbers summing to 100?

"3SUM" are there 3 numbers summing to 100?

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

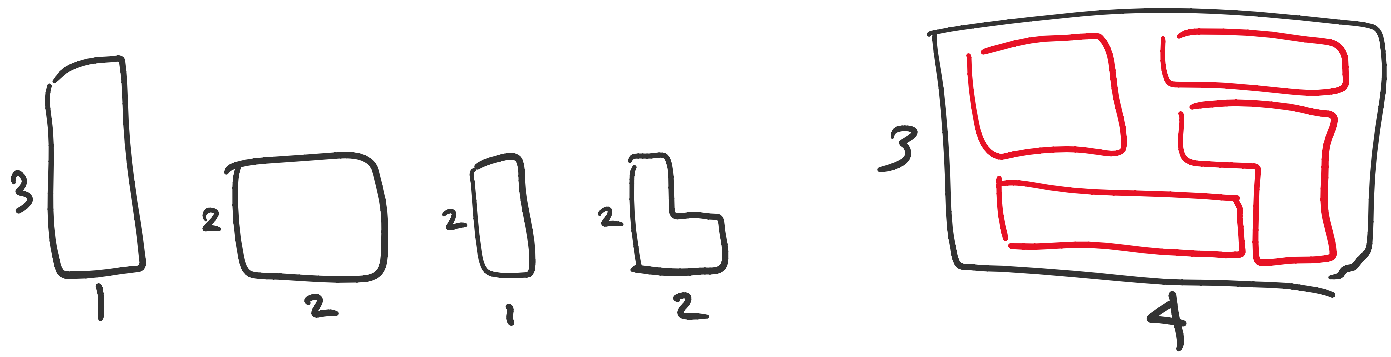
brute-force alg'm: $O(n^3)$ time

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

fastest? OPEN!

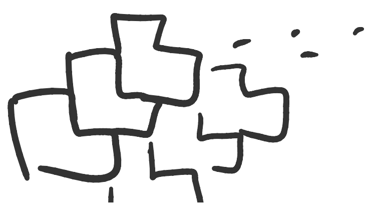
current record: $\tilde{O}\left(\frac{n^2}{\log^2 n}\right)$ (c. 2018)

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



no efficient alg'm believed possible
(NP-complete)

Ex3 Given n polygons (infinite # copies),
can they tile entire plane?



no alg'm possible (undecidable)

PART I. MODELS OF COMPUTATION

Math Prelims

Strings

A <u>string</u> is a finite sequence of symbols from a finite set Σ

can encode inputs to programs

called alphabet

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

0110, 10, 000, 0

ε denotes the empty string

Let Σ^* = {all strings over Σ}

Let x, y be strings.

a) length $|x|$

e.g. $|0110| = 4$, $|\varepsilon| = 0$.

b) <u>Concatenation</u> xy

e.g. $x = 10$, $y = 011$ $\Rightarrow$ $xy = 10011$

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

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

$\varepsilon x = x \varepsilon = x$

$\vdots$

$\varepsilon x = x\varepsilon = \ldots$

c) <u>ith power</u> $x^i = \underbrace{x \cdot \ldots \cdot x}_{i \text{ times}}$

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

$x^0 = \varepsilon$

d) x is a <u>substring</u> of y if

$y = wxz$ for some strings w, z

(<u>prefix</u> if $w = \varepsilon$, <u>suffix</u> if $z = \varepsilon$)

e) other ops:

x^R = reverse of x

(can be defined recursively:

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

$(xy)^R = y^R x^R$ (Lab 1a)

⋮

<u>Languages</u>

A <u>language</u> is a set L of strings

(i.e. $L \subseteq \Sigma^*$)

e.g. $\{0110, 101, 000, 0\}$

{ all words in English dictionary }

(... $\Sigma = \{$'a' ... 'z'$\}$)

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

infinite, more interesting

$\{ x \in \{0,1\}^* : |x| \text{ is even} \}$

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

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

$\vdots$

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. $L_1 = \{0, 00\}$. $L_2 = \{1, 01\}$

$\Rightarrow L_1 L_2 = \{ 01, 001, 0001 \}$

e.g. $L_1 = \{ 0, 00, 000, \ldots \} = \{ 0^i : i \geq 1 \}$

$L_2 = \{ 1, 11, 111, \ldots \} = \{ 1^i : i \geq 1 \}$

$\Rightarrow L_1 L_2 = \{ 0^i 1^j : i \geq 1, j \geq 1 \}$

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 star</u>

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

$$= L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots$$

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

$\{1, 01\}^* = \{\varepsilon, 1, 01, 11, 101, 011, 0101, 111, 101, 10101, 1101, \ldots\}$

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

(proof?)

$\{0,1\}^*$