SECTION 8.1
ALGORITHMS

ALGORITHMS

Algorithm: A clearly specified method (or procedure for solving a problem.

Etymology: Abu Ja'far Muḥammad ibn Mûsâ al-Khwârizmî

PROBLEMS

We distinguish between a problem and an instance of a problem.

PROBLEM	INSTANCE
Matrix multiplication	$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$
Traveling salesman	What is the most efficient route for my mail carrier?
Sudoku	

COMPLEXITY

Given an algorithm, we can ask about its cost.

The cost will be a function of the size of the input.
e.g. multiplying two numbers.

Complexity $f: \mathbb{N} \longrightarrow \mathbb{R}$

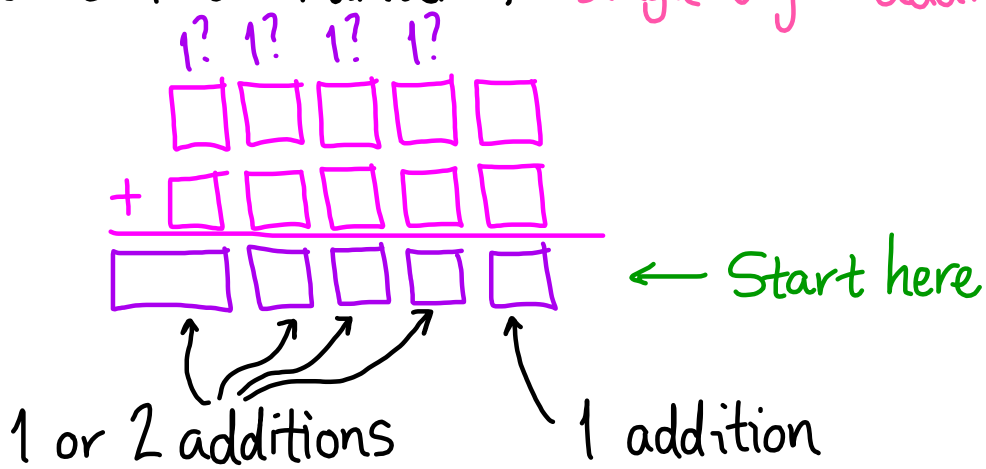
size of input $\rightsquigarrow$ cost for running the algorithm on input of that size. (worst case scenario)

For this to make sense, we need to specify what we mean by size and cost. Our answer will depend on the particular problem, as well as our particular needs.

COMPLEXITY

Example: What is the cost of adding two n digit numbers in terms of the number of single digit additions?

Answer:



$$\rightsquigarrow f(n) = 1 + 2(n-1) = 2n-1$$

THE BIG QUESTION

Is there a better way to do things?

Given two algorithms, which is more efficient?

Given one algorithm for a problem, is there a better one out there?

What do we mean by better?

$$7n^2 - 52 \sim n^2$$

but $\quad n < n^2$

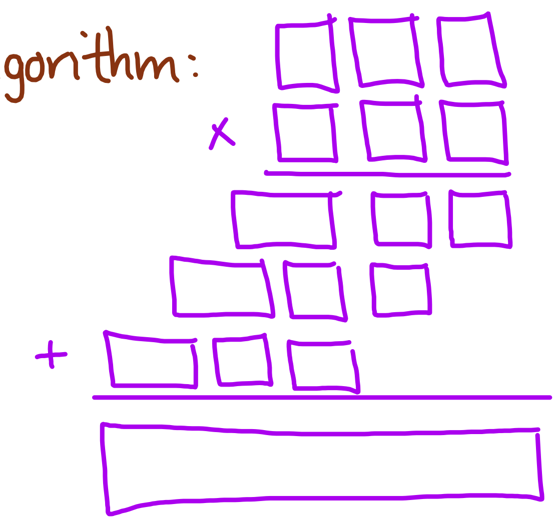
better than

EXAMPLE: MULTIPLYING TWO NUMBERS

Problem: What is the complexity of multiplying two n digit numbers in terms of the number of single digit multiplications?

(What about additions??)

Grade school algorithm:



$f(n) = n^2$

Is there a better way?

DIVIDE AND CONQUER

Idea: Break the problem into more manageable subproblems.

This usually leads to a recursive relation for the complexity, since the complexity of the bigger problem is given in terms of the complexity of the smaller problems.

Is there such an algorithm for multiplying two numbers?

MULTIPLYING TWO NUMBERS

Divide and Conquer Algorithm: The idea is to break up both n digit numbers into two $n/2$ digit numbers and multiply those:

$$a = \boxed{a_1 \mid a_2} = a_1 \cdot 10^{n/2} + a_2$$
$$b = \boxed{b_1 \mid b_2} = b_1 \cdot 10^{n/2} + b_2$$

$$ab = a_1 b_1 10^n + [a_1 b_2 + a_2 b_1]\, 10^{n/2} + a_2 b_2$$

Example: $1011 \cdot 1213 = (10 \cdot 12) \cdot 10^4 + (130 + 132) \cdot 10^2 + 143$
$= 1200000 + 26200 + 143 = 1226343.$

Complexity: $f(n) = 4 f(n-1)$

Actually, can improve from 4 to 3, since $(a_1 b_2 + a_2 b_1) = (a_1 + a_2)(b_1 + b_2) - a_1 b_1 - a_2 b_2$

$\rightsquigarrow f(n) = 3 f(n-1).$

MULTIPLYING TWO NUMBERS

So we find the complexity of the divide and conquer algorithm by solving the recurrence relation

$$a_n = 3a_{n/2}$$

We solve this recursion relation by working backward (see last page of Lecture 3).

Assume here $n = 2^k$: $a_n = 3a_{n/2} = 3^2 a_{n/4} = \cdots = 3^k a_1 = 3^k$

$$\rightsquigarrow \quad a_n = 3^{\log_2 n} = n^{\log_2 3}$$

This is <u>better</u> than n^2!

COMPARISON

#digits	# multiplications: Grade school algorithm	# multiplications: Divide & conquer algorithm	factor
10	100	39	2.5
100	10,000	1479	6.76
1,000	1,000,000	56,871	17.58
10,000	100,000,000	2,187,007	45.72
100,000	10,000,000,000	84,103,197	118.90
1,000,000	1×10^{12}	3,234,260,557	309.19

MORE EXAMPLES

① Matrix multiplication — Usual algorithm: n^2 multiplications
Divide & conquer: $n^{\log_2 7}$

Idea: Divide matrices into 4 submatrices, then do 7 multiplications.

② Evaluation of polynomials — Usual algorithm: $2n-1$ multiplications

Horner's method: Write $p(x)$ as $x \cdot q(x) + c$

$\rightsquigarrow$ recursive relation for # of multiplications:

$f(n) = f(n-1) + 1 \rightsquigarrow f(n) = n$

③ Greatest common divisor

④ Searching a list

A MILLION DOLLAR PROBLEM

A problem is of type P if it has a polynomial solution.

A problem is of type NP if, handed a solution to an instance of the problem, there is a polynomial time algorithm to check if it really is a solution.

e.g. factoring

QUESTION: P = NP?

> If P = NP, then the world would be a profoundly different place than we usually assume it to be. There would be no special value in "creative leaps," no fundamental gap between solving a problem and recognizing the solution once it's found. Everyone who could appreciate a symphony would be Mozart; everyone who could follow a step-by-step argument would be Gauss...
> — Scott Aaronson, MIT

NP-COMPLETE PROBLEMS

There is a (huge!) list of NP problems that "contain" all other NP problems, including:

Sudoku	Minesweeper
Battleship	Free Cell
Tetris	etc.

To prove P=NP, show any one of these problems has a polynomial solution.

To prove P≠NP, show any one of these problems has no polynomial solution.