Design and Analysis of Algorithms I

Introduction

Why Study Algorithms?

Why Study Algorithms?

- important for all other branches of computer science

Why Study Algorithms?

- important for all other branches of computer science
- plays a key role in modern technological innovation

Why Study Algorithms?

- important for all other branches of computer science
- plays a key role in modern technological innovation
  - "Everyone knows Moore's Law – a prediction made in 1965 by Intel co-founder Gordon Moore that the density of transistors in integrated circuits would continue to double every 1 to 2 years....in many areas, performance gains due to improvements in algorithms have vastly exceeded even the dramatic performance gains due to increased processor speed."
    - Excerpt from *Report to the President and Congress: Designing a Digital Future,* December 2010 (page 71).

Why Study Algorithms?

- important for all other branches of computer science
- plays a key role in modern technological innovation
- provides novel “lens” on processes outside of computer science and technology
  - quantum mechanics, economic markets, evolution

Why Study Algorithms?

- important for all other branches of computer science
- plays a key role in modern technological innovation
- provides novel “lens” on processes outside of computer science and technology
- challenging (i.e., good for the brain!)

Why Study Algorithms?

- important for all other branches of computer science
- plays a key role in modern technological innovation
- provides novel "lens" on processes outside of computer science and technology
- challenging (i.e., good for the brain!)
- fun

Integer Multiplication

Input: 2 n-digit numbers x and y.

Output: product $x \cdot y$.

"Primitive Operation": add or multiply 2 single-digit numbers.

The Grade-School Algorithm

$$\begin{array}{r} 5678 \\ \times\ \ 1234 \\ \hline 22712 \\ 17034\,- \\ 11356\,-\,- \\ 5678\,-\,-\,- \\ \hline 7006652 \end{array}$$

roughly n operations per row (up to a constant)

$\#$ operations overall $\approx$ constant $\cdot\, n^2$

The Algorithm Designer's Mantra

"Perhaps the most important principle for the good algorithm designer is to refuse to be content."

-Aho, Hopcroft, and Ullman, *The Design and Analysis of Computer Algorithms*, 1974

CAN WE DO BETTER?
[than the "obvious" method]

A Recursive Algorithm

Write $x = 10^{n/2} a + b$ and $y = 10^{n/2} c + d$
where a, b, c, d are $\frac{n}{2}$-digit numbers.

[example: $a = 56$, $b = 78$, $c = 12$, $d = 34$]

Then $x \cdot y = (10^{n/2} a + b)(10^{n/2} c + d)$
$$= 10^n ac + 10^{n/2}(ad + bc) + bd \qquad (*)$$

Idea: recursively compute ac, ad, bc, bd, then compute $(*)$ in the obvious way.

(simple base case omitted)

Karatsuba Multiplication

$x \cdot y = 10^n ac + 10^{n/2}(ad+bc) + bd$

① Recursively compute $\cancel{ac}$

② Recursively compute $\cancel{bd}$

③ Recursively compute $(a+b)(c+d) = \cancel{ac} + \cancel{bd} + ad + bc$

Gauss's trick : ③ − ① − ② = $ad + bc$

Upshot: only need 3 recursive multiplications (and some additions).

Q: which is the fastest algorithm?

Tim Roughgarden