Integer addition and subtraction

Addition. Given two n-bit integers a and b, compute $a + b$.

Subtraction. Given two n-bit integers a and b, compute $a - b$.

Grade-school algorithm. $\Theta(n)$ bit operations. ← "bit complexity" (instead of word RAM)

1	1	1	1	1	1	0	1	
	1	1	0	1	0	1	0	1
+	0	1	1	1	1	1	0	1
1	0	1	0	1	0	0	1	0

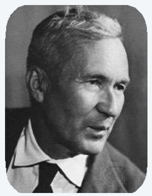
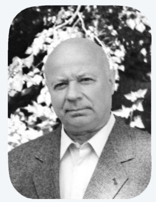
Remark. Grade-school addition and subtraction algorithms are optimal.

Integer multiplication

Multiplication. Given two n-bit integers a and b, compute $a \times b$.

Grade-school algorithm (long multiplication). $\Theta(n^2)$ bit operations.

									1	1	0	1	0	1	0	1
								×	0	1	1	1	1	1	0	1
									1	1	0	1	0	1	0	1
								0	0	0	0	0	0	0	0	
							1	1	0	1	0	1	0	1		
						1	1	0	1	0	1	0	1			
					1	1	0	1	0	1	0	1				
				1	1	0	1	0	1	0	1					
			1	1	0	1	0	1	0	1						
		0	0	0	0	0	0	0	0							
	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1

Conjecture. [Kolmogorov 1956] Grade-school algorithm is optimal.

Theorem. [Karatsuba 1960] Conjecture is false.

Divide-and-conquer multiplication

To multiply two n-bit integers x and y:

- Divide x and y into low- and high-order bits.
- Multiply four ½n-bit integers, recursively.
- Add and shift to obtain result.

$$m = \lceil n / 2 \rceil$$

$$a = \lfloor x / 2^m \rfloor \qquad b = x \bmod 2^m$$

$$c = \lfloor y / 2^m \rfloor \qquad d = y \bmod 2^m$$

use bit shifting to compute 4 terms

$$x\,y = (2^m\, a + b)\,(2^m\, c + d) = 2^{2m}\, ac + 2^m\,(bc + ad) + bd$$

Ex. $x = \underbrace{1\,0\,0\,0\,1}_{a}\ \underbrace{1\,0\,1}_{b} \qquad y = \underbrace{1\,1\,1}_{c}\ \underbrace{0\,0\,0\,0\,1}_{d}$

Divide-and-conquer multiplication

MULTIPLY(x, y, n)

IF $(n = 1)$

  RETURN x ! y.

ELSE

  $m \leftarrow \lceil n / 2 \rceil$.

  $a \leftarrow \lfloor x / 2^m \rfloor$; $b \leftarrow x \bmod 2^m$.

  $c \leftarrow \lfloor y / 2^m \rfloor$; $d \leftarrow y \bmod 2^m$. ← $\Theta(n)$

  $e \leftarrow$ MULTIPLY(a, c, m).

  $f \leftarrow$ MULTIPLY(b, d, m).

  $g \leftarrow$ MULTIPLY(b, c, m).

  $h \leftarrow$ MULTIPLY(a, d, m). ← $4\,T(\lceil n/2 \rceil)$

  RETURN $2^{2m}\, e + 2^m\, (g + h) + f$. ← $\Theta(n)$

What's the complexity?

Karatsuba trick

To multiply two n-bit integers x and y:

- Divide x and y into low- and high-order bits.
- To compute middle term $bc + ad$, use identity:

$$bc + ad = ac + bd - (a - b)(c - d)$$

➔ Multiply only three ½n-bit integers, recursively.

$$m = \lceil n / 2 \rceil$$

$$a = \lfloor x / 2^m \rfloor \qquad b = x \bmod 2^m$$

$$c = \lfloor y / 2^m \rfloor \qquad d = y \bmod 2^m$$

$$x = \underbrace{1\,0\,0\,0\,1}_{a}\ \underbrace{1\,0\,1}_{b}$$

$$y = \underbrace{1\,1\,1}_{c}\ \underbrace{0\,0\,0\,0\,1}_{d}$$

middle term

$$x\,y = (2^m a + b)(2^m c + d) = 2^{2m} ac + 2^m (bc + ad) + bd$$

$$= 2^{2m} ac + 2^m (ac + bd - (a - b)(c - d)) + bd$$

1 1 3 2 3

Karatsuba multiplication

KARATSUBA-MULTIPLY(x, y, n)

IF $(n = 1)$

  RETURN x ! y.

ELSE

  $m \leftarrow \lceil n / 2 \rceil$

  $a \leftarrow \lfloor x / 2^m \rfloor$; c $b \leftarrow x \bmod 2^m$.

  $\leftarrow \lfloor y / 2^m \rfloor$; $e \leftarrow$ $d \leftarrow y \bmod 2^m$.

  ← $\Theta(n)$

  $f \leftarrow$ KARATSUBA-MULTIPLY(a, c, m).

  $g \leftarrow$ KARATSUBA-MULTIPLY(b, d, m).

  KARATSUBA-MULTIPLY$(|a - b|, |c - d|, m)$

  ← $3\,T(\lceil n / 2 \rceil)$

  Flip sign of g if needed.

  RETURN $2^{2m}\, e + 2^m\, (e + f - g) + f$.   ← $\Theta(n)$

Karatsuba analysis

Proposition. Karatsuba's algorithm requires $O(n^{1.585})$ bit operations to multiply two n-bit integers.

Pf. Apply Case 1 of the master theorem to the recurrence:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 3T(\lceil n/2 \rceil) + \Theta(n) & \text{if } n > 1 \end{cases}$$

$$\Longrightarrow T(n) = \Theta\left(n^{\log_2 3}\right) = O\left(n^{1.585}\right)$$

Practice.

- Use base 32 or 64 (instead of base 2).
- Faster than grade-school algorithm for about 320–640 bits.