The Design and Analysis of Algorithms

Lecture 9 Divide and Conquer II

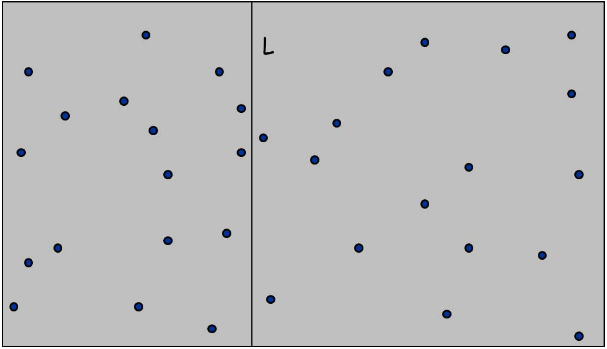
Zhenbo Wang

Department of Mathematical Sciences, Tsinghua University

Content

Closest Pair of Points: Second Attempt

- *Divide*: draw vertical line L so that roughly $n/2$ points on each side.
- *Conquer*: find closest pair in each side recursively.
- *Combine*: find closest pair with one point in each side.
- Return best of 3 solutions.



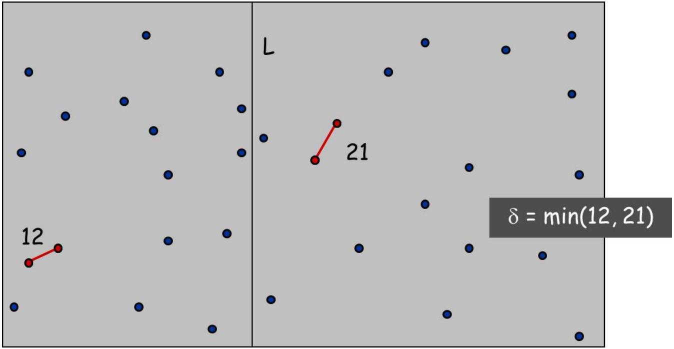
Find Closest Pair with One Point in Each Side

- Find closest pair with one point in each side, assuming that distance $< \delta$.

  Observation: only need to consider points within δ of line L.

  Sort points in 2δ-strip by their y coordinate.

  Only check distances of those within 11 positions in sorted list!



Find Closest Pair with One Point in Each Side

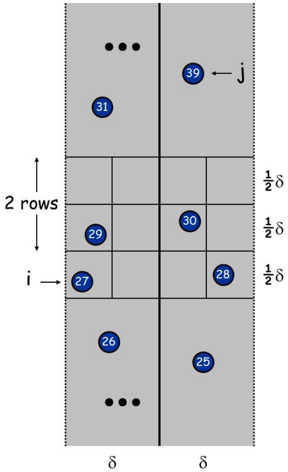
Def. Let s_i be the point in the 2δ-strip, with the i^{th} smallest y-coordinate.

Claim. If $|i - j| \geq 12$, then the distance between s_i and s_j is at least δ.

Pf. No two points lie in same $\frac{\delta}{2}$-by-$\frac{\delta}{2}$ box.

Two points at least 2 rows apart have distance $\geq 2\frac{\delta}{2}$.

Fact. Still true if we replace 12 with 7.



Closest Pair: Divide-and-conquer Algorithm

$CLOSEST - PAIR(p_1, p_2, \cdots, p_n)$

1. Compute separation line L such that half the points are on each side of the line. $O(n \log n)$
2. $\delta_1 \leftarrow$ CLOSEST-PAIR (points in left half).
3. $\delta_2 \leftarrow$ CLOSEST-PAIR (points in right half). $2T(n/2)$
4. $\delta \leftarrow \min\{\delta_1, \delta_2\}$.
5. Delete all points further than δ from line L. $O(n)$
6. Sort remaining points by y-coordinate. $O(n \log n)$
7. Scan points in y-order and compare distance between each point and next 11 neighbors. If any of these distances is less than δ, update δ. $O(n)$
8. **return** δ.

Closest Pair of Points: Analysis

- Running time.

$$T(n) \leq 2T(n/2) + O(n \log n) \Rightarrow T(n) = O(n \log^2 n).$$

Q. Can we achieve $O(n \log n)$?

A. Yes. Don't sort points in strip from scratch each time.

Each recursive returns two lists: all points sorted by y coordinate, and all points sorted by x coordinate.

Sort by merging two pre-sorted lists.

$$T(n) \leq 2T(n/2) + O(n) \Rightarrow T(n) = O(n \log n).$$

Master Method

Goal. Recipe for solving common divide-and-conquer recurrences:

$$T(n) = aT(\frac{n}{b}) + f(n).$$

Terms. $a \geq 1$ is the number of subproblems.

$b > 0$ is the factor by which the subproblem size decreases.

$f(n) =$ work to divide/merge subproblems.

- *Recursion tree*.

$k = \log_b n$ levels.

$a^i =$ number of subproblems at level i.

$n/b^i =$ size of subproblem at level i.

Master Theorem

Theorem 1 (Master Theorem)

Suppose that $T(n)$ is a function on the nonnegative integers that satisfies the recurrence

$$T(n) = aT(\frac{n}{b}) + f(n)$$

where $\frac{n}{b}$ means either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Let $k = log_b a$,

Case 1. *If $f(n) = O(n^{k-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^k)$.*

Case 2. *If $f(n) = \Theta(n^k \log^p n)$, then $T(n) = \Theta(n^k \log^{p+1} n)$.*

Case 3. *If $f(n) = \Omega(n^{k+\epsilon})$ for some constant $\epsilon > 0$ and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.*

Integer Arithmetic

- *Addition*. Given two n-bit integers a and b, compute $a + b$.
- *Subtraction*. Given two n-bit integers a and b, compute $a - b$.
- *Multiplication*. Given two n-bit integers a and b, compute $a \times b$.
- *Grade-school algorithm*: $\Theta(n)$ bit operations for addition.
- *Remark*. Grade-school addition and subtraction algorithms are asymptotically optimal.

	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

Integer Multiplication

- *Grade-school algorithm* for 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 1 (Kolmogorov 1952)

Grade-school algorithm is optimal.

Theorem 2 (Karatsuba 1960)

Conjecture is wrong.

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 $n/2$-bit integers, recursively.

  Add and shift to obtain result.

$$
\begin{aligned}
& m = \lceil n/2 \rceil \\
& a = \lfloor x/2^m \rfloor, \quad b = x \pmod{2^m} \\
& c = \lfloor y/2^m \rfloor, \quad d = y \pmod{2^m}
\end{aligned}
$$

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

Ex. $x = 10001101, y = 11100001$:
$a = 1000, b = 1101, c = 1110, d = 0001.$

Divide-and-Conquer Multiplication

MULTIPLY(x, y, n)

if $n = 1$ **then**

    return $x \times y$.

else

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

    $a \leftarrow \lfloor x/2^m \rfloor$; $b \leftarrow x \pmod{2^m}$.

    $c \leftarrow \lfloor y/2^m \rfloor$; $d \leftarrow y \pmod{2^m}$.

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

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

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

    $h \leftarrow MULTIPLY(a, d, m)$.

    return $2^{2m}e + 2^m(g + h) + f$.

end if

Divide-and-Conquer Multiplication: Analysis

- *Proposition*. The divide-and-conquer multiplication algorithm requires $\Theta(n^2)$ bit operations to multiply two n-bit integers.

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

$$T(n) = 4T(n/2) + \Theta(n) \Rightarrow T(n) = \Theta(n^2).$$

- Karatsuba trick:

  To compute middle term $bc + ad$, use identity
  $bc + ad = ac + bd - (a - b)(c - d)$.

$$\begin{aligned} & (2^m a + b)(2^m c + d) = 2^{2m} ac + 2^m (bc + ad) + bd \\ = \; & 2^{2m} ac + 2^m (ac + bd - (a - b)(c - d)) + bd. \end{aligned}$$

  Only three multiplication of $n/2$-bit integers, say ac, bd and $(a - b)(c - d)$.

Karatsuba Multiplication

KARATSUBA – MULTIPLY(x, y, n)

if $n = 1$ **then**
  return $x \times y$.
else
  $m \leftarrow \lceil n/2 \rceil$.
  $a \leftarrow \lfloor x/2^m \rfloor$; $b \leftarrow x \pmod{2^m}$.
  $c \leftarrow \lfloor y/2^m \rfloor$; $d \leftarrow y \pmod{2^m}$.
  $e \leftarrow KARATSUBA - MULTIPLY(a, c, m)$.
  $f \leftarrow KARATSUBA - MULTIPLY(b, d, m)$.
  $g \leftarrow KARATSUBA - MULTIPLY(a - b, c - d, m)$.
  return $2^{2m}e + 2^m(e + f - g) + f$.
end if

Karatsuba Multiplication: 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:

$$T(n) = 3T(n/2) + \Theta(n) \Rightarrow T(n) = \Theta(n^{\log 3}) = O(n^{1.585}).$$

History of Asymptotic Complexity of Integer Multiplication

year	algorithm	order of growth
?	brute force	$\Theta(n^2)$
1962	Karatsuba-Ofman	$\Theta(n^{1.585})$
1963	Toom-3, Toom-4	$\Theta(n^{1.465}), \Theta(n^{1.404})$
1966	Toom-Cook	$\Theta(n^{1+\epsilon})$
1971	Schönhage-Strassen	$\Theta(n \log n \log \log n)$
2007	Fürer	$n \log n 2^{O(\log^* n)}$
2019	Harvey-van der Hoeven	$O(n \log n)$
?	?	$\Theta(n)$

Homework

- Read Chapter 5 of the textbook.
- Prove the Master Theorem.