CS 580
ALGORITHM DESIGN AND ANALYSIS

Divide and Conquer 2:
Closest Pairs & Multiplication

Vassilis Zikas

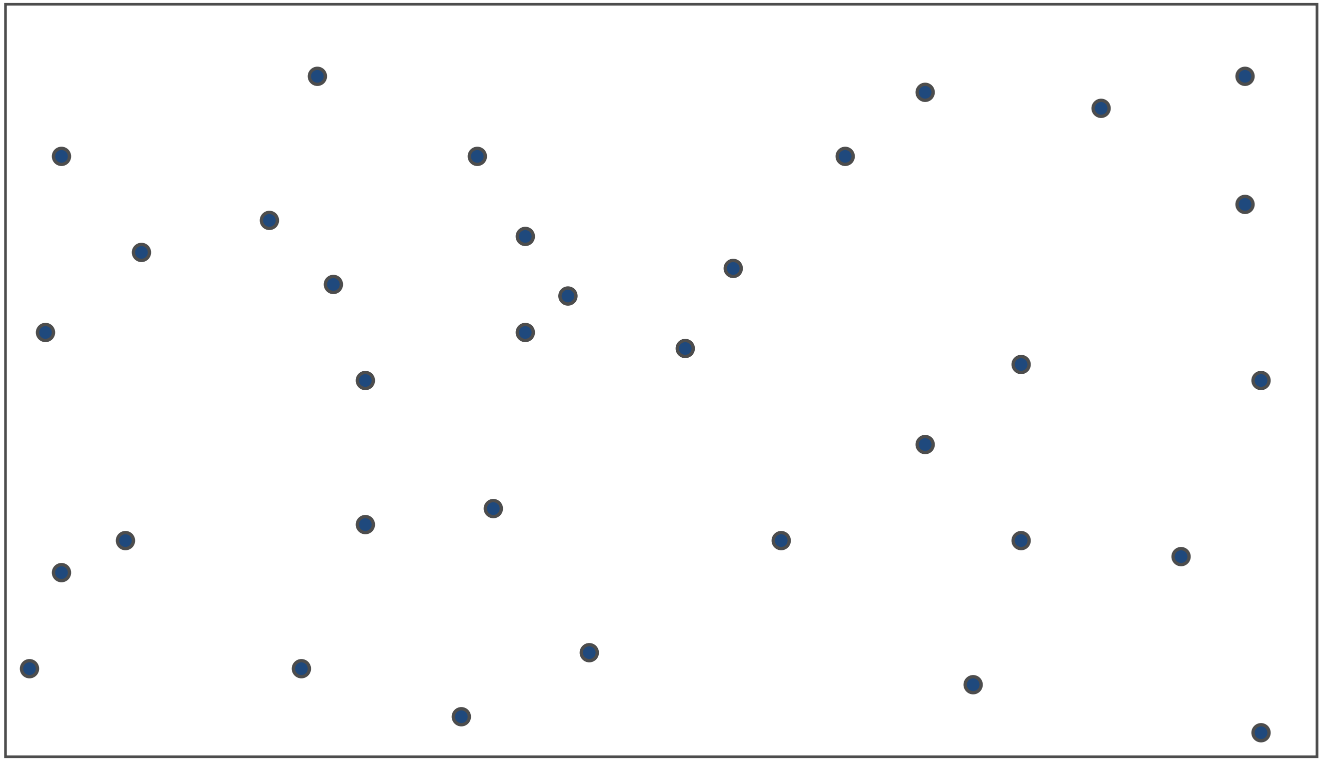
PLAN

- Closest Pair of Points (5.4 in KT)
- Integer Multiplication (5.5 in KT)
- Matrix Multiplication (4.2 in CLRS)x

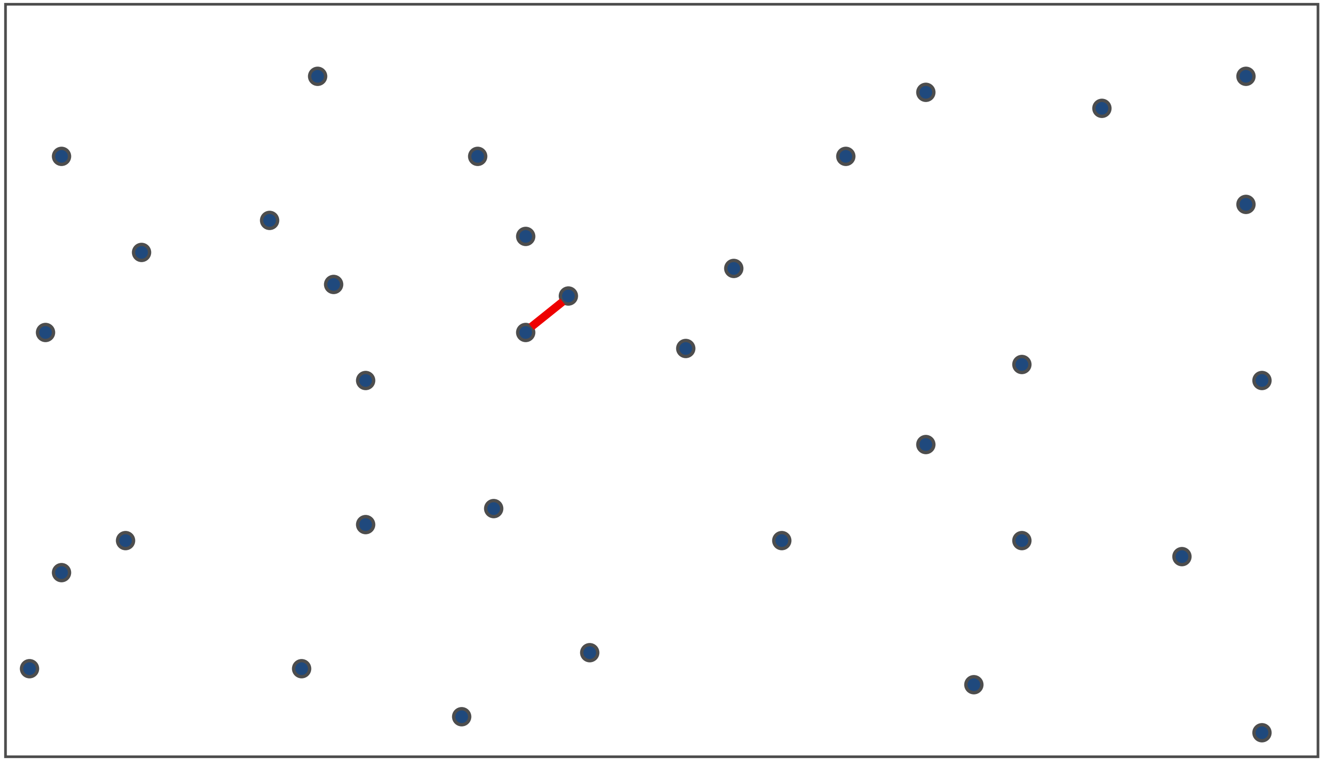
CLOSEST PAIR OF POINTS

- Input: n points in two dimensions
  - $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
  - Simplifying assumption: no two points have the same x-coordinate
- Output:
  - The pair of points $(x_i, y_i), (x_j, y_j)$ with the smallest Euclidean distance

CLOSEST PAIR OF POINTS



CLOSEST PAIR OF POINTS

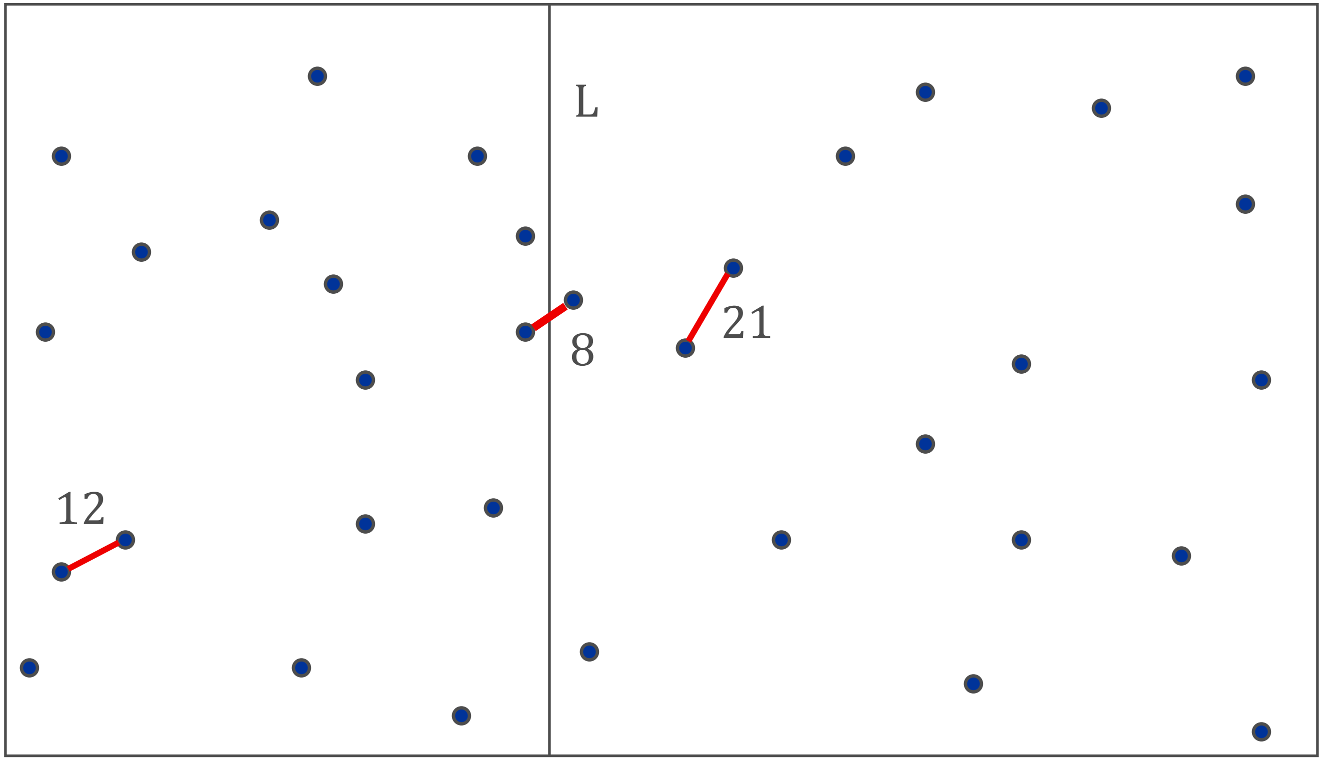


CLOSEST PAIR OF POINTS

- Brute force: $O(n^2)$
- 1-D version:
  - Sort the points/numbers
  - Move left to right remembering the closest pair you've seen so far
- 2-D version: Maybe sort by y_i or x_i?
  - Very easy to see that close in x or y doesn't imply anything about Euclidean distance
- Maybe mergesort type of algorithm?
  - Split into two inputs

CLOSEST PAIR OF POINTS

- Divide: split so that roughly $n/2$ points in each side
- Conquer: find closest pair in each side, recursively
- Combine: find closest pair with one point in each side

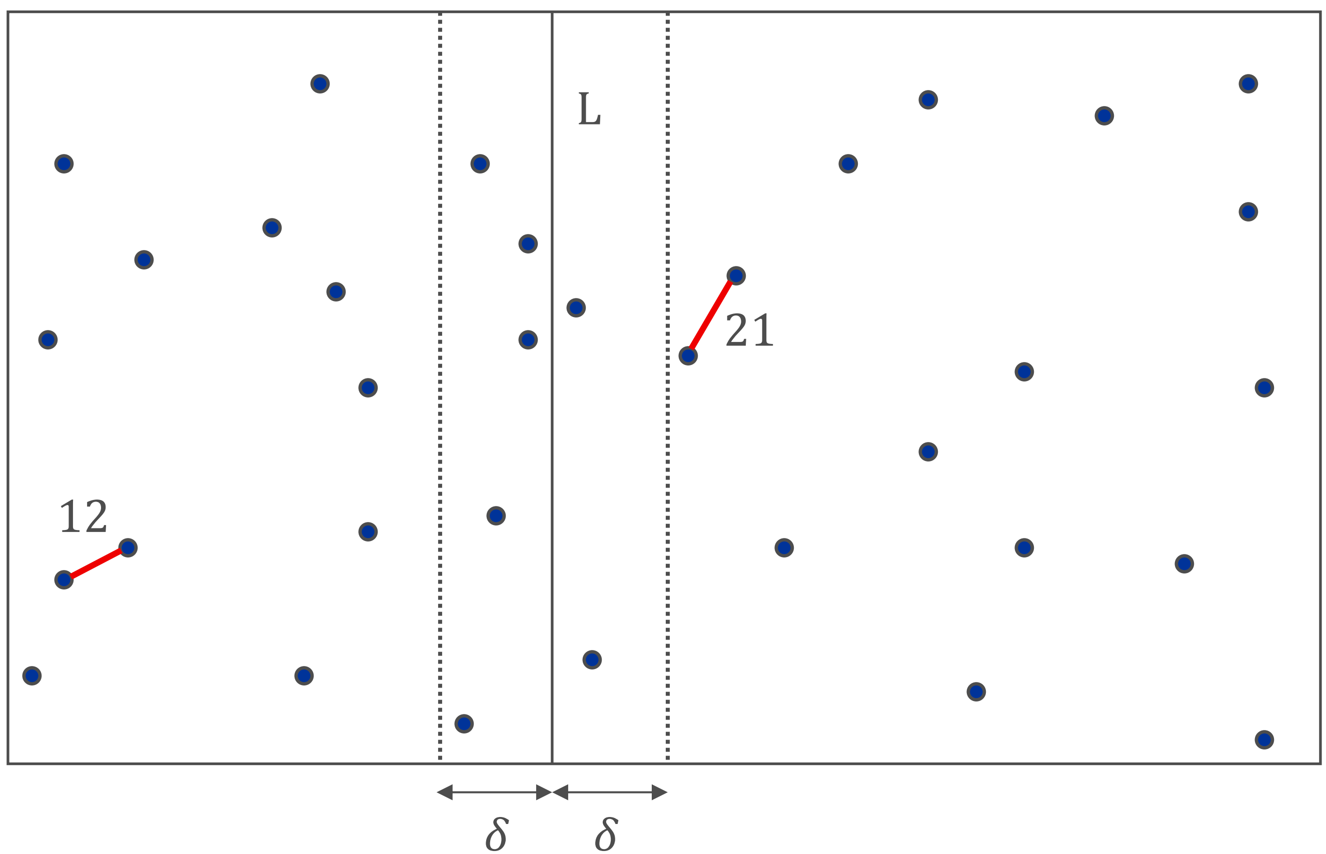


CLOSEST PAIR OF POINTS

- Combine feels like it should take $\Theta(n^2)$
- Trick: don't look too far from L

CLOSEST PAIR OF POINTS

- No reason to look further than $\delta = \min(12{,}21)$
- What if all the points are in this band??

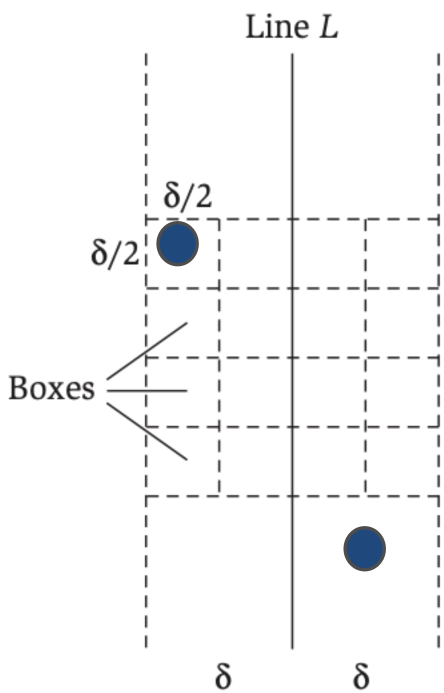


CLOSEST PAIR OF POINTS

- Let S be the set of points in the band, and let S_y be the points in the band sorted by y_i
- Claim: If $s, s' \in S$ are such that $d(s, s') < \delta$, then s and s' are within 15 (!!!!) positions of each other in the sorted list S_y
- Proof:
  - Break Z, the subset of the plane of distance δ from L into squares of side $\delta/2$

CLOSEST PAIR OF POINTS

- There can't be two points in the same square!
  - The points would be on the same side and their distance would be at most $\frac{\delta\sqrt{2}}{2} < \delta$!



Separated by at least 3 rows, therefore distance at least $3\delta/2$

CLOSEST PAIR OF POINTS

- Overall algorithm:
  - Find L that splits points into two sets with $n/2$
    - How? Sort by x and pick median. $O(nlog(n))$
  - $\delta_1 = Closest\_Pair(left)$
  - $\delta_2 = Closest\_Pair(right)$
  - $\delta = \min(\delta_1, \delta_2)$
  - Delete all points further than δ from L: $O(n)$
  - Sort remaining points by y: $O(nlog(n))$
  - Scan in y-order, comparing each point to the 15 points ahead of it; update δ as you go: $O(n)$

CLOSEST PAIR OF POINTS

- Running time:
  - $T(n) = 2T\left(\frac{n}{2}\right) + O(nlogn)$
  - Hmmmm… Doesn't fit Master Theorem the way we've seen it
  - But, notice that the two times we sort can be done in the beginning!
  - The rest is linear
  - Overall: $\Theta(nlogn) + T(n)$, where $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$
    - So, $O(nlogn)$ in total

MULTIPLICATION!

INTEGER MULTIPLICATION

- Input: two n-bit numbers, x and y
- Output: their product xy

INTEGER MULTIPLICATION

- Grade school algorithm: $O(n^2)$

$$
\begin{array}{r}
12 \\
\times\, 13 \\
\hline
36 \\
12 \\
\hline
156
\end{array}
$$

Decimal

$$
\begin{array}{r}
1100 \\
\times\, 1101 \\
\hline
1100 \\
0000 \\
1100 \\
1100 \\
\hline
10011100
\end{array}
$$

Binary

INTEGER MULTIPLICATION

- Grade school algorithm: $O(n^2)$
- Each line is a partial product
- Add up all the partial products
- $O(n)$ time to compute each one
- $O(n)$ time to add them all up
- Isn't all this necessary??

```
     1100
   × 1101
   ------
     1100
    0000
   1100
  1100
 --------
 10011100
```

INTEGER MULTIPLICATION

- I guess we're supposed to divide and conquer
- Split x and y into two $n/2$ bit numbers
  - $x = 2^{\frac{n}{2}} x_L + x_R$
  - $y = 2^{\frac{n}{2}} y_L + y_R$
- E.g.: $x = 10110110$
  - $x_L = 1011$
  - $x_R = 0110$
  - $x = 1011 \cdot 2^4 + 0110 = 10110000 + 0110$

INTEGER MULTIPLICATION

- $x \cdot y = (x_L \cdot 2^{\frac{n}{2}} + x_R)(y_L \cdot 2^{\frac{n}{2}} + y_R)$
- $= x_L y_L 2^n + x_L y_R 2^{\frac{n}{2}} + x_R y_L 2^{\frac{n}{2}} + x_R y_R$
- Overall, 4 subproblems of size $n/2$
- Linear time to merge
- $T(n) = 4T\left(\frac{n}{2}\right) + O(n)$
- $a = 4, b = 2$: $\log_b a = 2$
- Master theorem says $T(n) \in O(n^2)$
- Oh well…

INTEGER MULTIPLICATION

Gauss (1777-1855)

- Tricks!!!
- Multiplication of complex numbers
$$(a + b \cdot i)(c + d \cdot i)$$
$$= ac - bd + (bc + ad) \cdot i$$
- But, $bc + ad = (a + b)(c + d) - ac - bd$
- Only three multiplications!
  - ac, bd and $(a + b)(c + d)$

INTEGER MULTIPLICATION

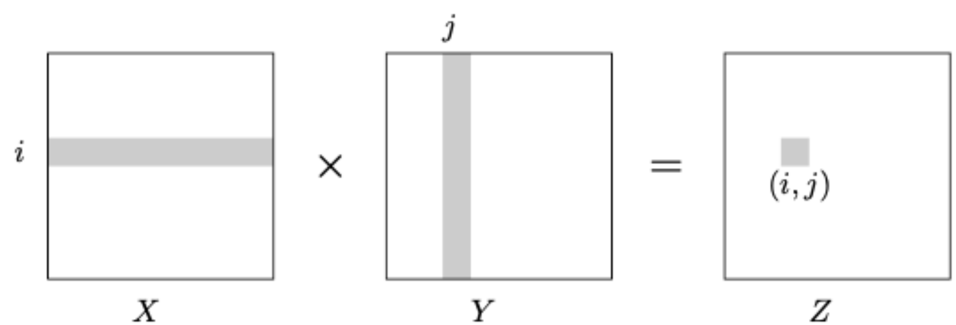
- Back to our problem
- $x \cdot y = x_L y_L 2^n + x_L y_R 2^{\frac{n}{2}} + x_R y_L 2^{\frac{n}{2}} + x_R y_R$
- $= x_L y_L 2^n + x_R y_R + 2^{\frac{n}{2}}(x_L y_R + x_R y_L)$

Gauss:

- $(x_L y_R + x_R y_L) = (x_L + x_R)(y_L + y_R) - x_L y_L - x_R y_R$
- Only three multiplications: $x_L y_L$, $x_R y_R$ and $(x_L + x_R)(y_L + y_R)$
- Master theorem: $a = 3, b = 2$
  - Runtime $O\left(n^{\log_b a}\right) = O(n^{1.59})$!

MATRIX MULTIPLICATION

- Input:
  - Two n by n matrices X and Y
- Output:
  - $Z = XY$
  - $Z_{ij} = \sum_{k=1}^{n} X_{ik} Y_{kj}$



MATRIX MULTIPLICATION

- Could we hope for better than $\Theta(n^3)$?
- Tricks!!
- Strassen (1969)

Volker Strassen

MATRIX MULTIPLICATION

- Multiplication by blocks
- $X = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, Y = \begin{bmatrix} E & F \\ G & H \end{bmatrix}$
- $XY = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$
- 8 subproblems of size $n/2$
- Merging (adding) takes $O(n^2)$
- $T(n) = 8T\left(\frac{n}{2}\right) + O(n^2)$
- Master theorem: $T(n) \in O(n^3)$…

MATRIX MULTIPLICATION

- Better algebra
- $XY = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$
  - $P_1 = A(F - H)$
  - $P_2 = (A + B)H$
  - $P_3 = (C + D)E$
  - $P_4 = D(G - E)$
  - $P_5 = (A + D)(E + H)$
  - $P_6 = (B - D)(G + H)$
  - $P_7 = (A - C)(E + F)$
- 7 subproblems of size $n/2$, with $O(n^2)$ merging time
- Master theorem: $T(n) \in O\left(n^{\log_2 7}\right) \approx O(n^{2.81})$

MATRIX MULTIPLICATION

- Faster??
- Multiplying two 2-by-2 matrices with 6 scalar multiplications is impossible [Hopcroft and Kerr 1971]
- Two 20-by-20 matrices with 4460 multiplications: $O(n^{2.805})$
- Two 48-by-48 matrices with 47217 multiplications: $O(n^{2.7801})$
- 1990: Coppersmith-Winograd $O(n^{2.376})$
- 2014: Williams $O(n^{2.372873})$
- 2014: Le Gall $O(n^{2.3728639})$
- 2020: Alman-Williams $O(n^{2.3728596})$
- Caveat: this is only worthwhile for matrices too big to fit in modern computers…

SUMMARY

- Closest Pair of Points
- Multiplication:
  - Integer multiplication
  - Matrix multiplication