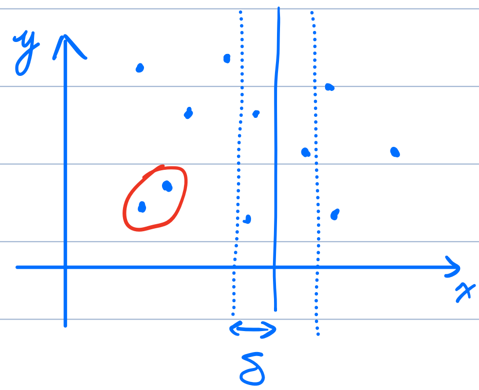
1. Closest Pair of Points (cont'd)

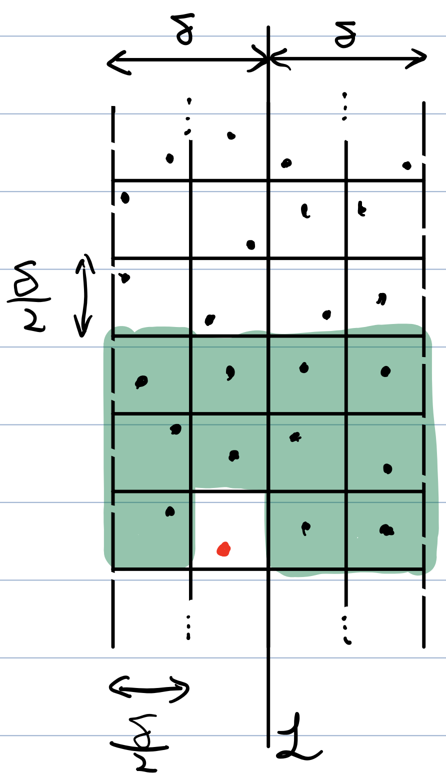
Input: n points on a plane $(x_1, y_1)(x_2, y_2) \dots (x_n, y_n)$

Goal: find two points that are closest to each other

Idea: Divide points into two halves, solve the subproblem for each half, then combine.

need to consider cross-border pairs





1. Sort the points near the border by their y coordinate
2. Overlay a $\frac{\delta}{2} \times \frac{\delta}{2}$ grid
3. Process the points according to their y coordinate

Claims:

1. At most 1 point per cell
2. Only need to look at points at most two rows away $\Rightarrow$ at most 11 cells! (with some careful argument, this can be 6)

Compute "cross-border" closest pair:

1. Take points within δ of L
2. Sort these points by y coordinate $p_1, p_2, \ldots, p_k$
3. Check the distance between each point p_i with the 11 points after it in the list

   For $i = 1$ to k:

   For $j = i+1$ to $i+11$:

   Check $\mathrm{dist}(p_i, p_j)$, and update min if needed.
4. Output the closest pair found

Correctness: ✓

Runtime:

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n \log n) \Rightarrow T(n) = \Theta(n \log^2 n)$$

Optimization:

Notice we repeated sort every time!

Sort once at the very beginning:

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n) \Rightarrow T(n) = \Theta(n \log n)$$

$(f(n) \downarrow)$

2. Median and Order Statistics

$$a_1 < a_2 < a_3 < \dots < a_n < \cdots < a_{2n-2} < a_{2n-1}$$

$\uparrow$ min, $\uparrow$ median, $\uparrow$ max

"n-1 elements to the left, n-1 to the right"

$$a_1 < \cdots < a_n < a_{n+1} < \cdots < a_{2n}$$

$\uparrow$ lower median, $\uparrow$ upper median

Depending on the scenario, median $= \frac{\text{upper}+\text{lower}}{2}$ / upper / lower

For this course, simply use lower median

Problem: Given unsorted $a_1, a_2, \dots, a_n$ (for simplicity, assume distinct).

Find i-th smallest element.

- $i = 1/n$: min/max

  Easy! $n-1$ comparisons, $\Theta(n)$

  What if want both min & max?

  Naive: $2(n-1)$ comparisons

  Better: Read two elements, compare them, then compare larger w/ max, and smaller w/ min $\Rightarrow \frac{3}{2}(n-1)$ comparisons

- $i=2/n-1$: Keep two variables min1, min2
  Still $\Theta(n)$
- $i=3/n-3$: Still $\Theta(n)$

  ⋮
- $i=k/n-k$?
- $i=\lceil \frac{n}{2} \rceil$? Median $\Theta(n^2)$? $\Theta(n\log n)$? $\Theta(n^{\log_2 3})$?
  $\Theta(n)$? $\Theta(\log n)$?

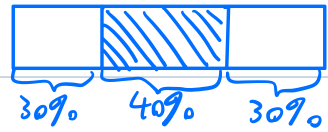
Finding the Median

Intuition: Use idea similar to QuickSort!

```
RandomizedSelect(A,i)
1.   pick j randomly from {1,2,..., len(A)} // pivot=A[j]
2.   k = Partition(A,j) // pivot is now at A[k]
3.   If k=i
4.       Return A[k]
5.   ElseIf k>i
6.       Return RandomizedSelect(A[1,...,k-1], i)
7.   Else // k<i
8.       Return RandomizedSelect(A[k+1,...,n], i-k)
9.   EndIf
```

Correctness: ✓ (perfect correctness)

Runtime Analysis (rough): We count on the pivot $A[j]$ to be "approx-median", i.e. not in top/bottom 30%, hence in the middle 40%. This happens with probability 40%.



So expected runtime is roughly:

$$T(n) = T\left(\tfrac{7}{10}n\right) + \Theta(n) \Rightarrow T(n) = \Theta(n)$$

Same as finding min/max!!

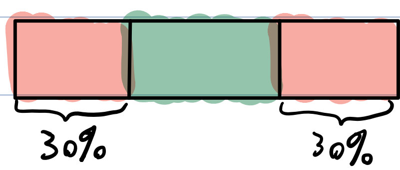
But it's randomized......

What if we want deterministic alg.?

Idea: Replace step 1 with some deterministic algorithm to find "approx-median"

First attempt:

Compute recursively the median x of $A[1, \ldots, \frac{3}{5}n]$



- How many elements smaller than x? $\geq 30\%$
- How many elements greater than x? $\geq 30\%$
- Must be in the middle 40%!

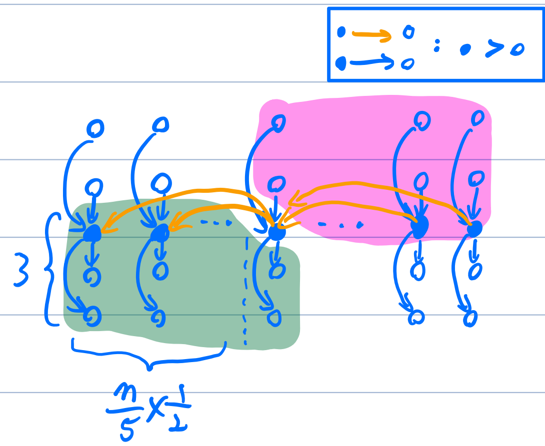
$$T(n) = T\left(\tfrac{3}{5}n\right) + T\left(\tfrac{7}{10}n\right) + \Theta(n)$$

$$\Rightarrow T(n) = \Omega(n \log n) \quad \text{(by recursion tree)}$$

More precisely, $T(n) = \Theta(n^{1.616})$ (by Akra-Bazzi Method) * not required for this course

Actual Algorithm (to find "approx-median"):

1. Partition A into $\frac{n}{5}$ sets of size 5 each.
2. Compute median of each set in $O(1)$.
3. Compute median of these $\frac{n}{5}$ medians, that'll be our "approx-median" x.



- How many elements smaller than x? # elements in green area $\geq (\frac{n}{5} \times \frac{1}{2}) \times 3 = \frac{3}{10}n$

- How many elements greater than x? # elems in pink area: $\geq \frac{3}{10}n$

Plugging this into RandomizedSelect, Runtime is now:

$$T(n) = \underline{T(\tfrac{n}{5})} + T(\tfrac{7}{10}n) + \Theta(n)$$

$$\Rightarrow T(n) = \Theta(n)$$

Same as finding min/max!!

Also deterministic!!

: Reduce the size of subproblems