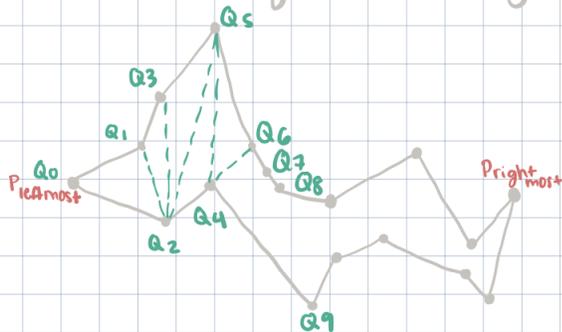


REGULARIZING A PLANAR SUBDIVISION

Regularizing a Planar Subdivision

Regularizing into monotone regions: $O(n \log n)$
 Splitting monotone polygon into triangular regions: $\Theta(n)$

- into monotone regions $\rightarrow O(n \log n)$
- monotone polygon into Δ regions \downarrow



◦ can get left & rightmost in $\log n$
 ◦ want sorted order by x-coordinate in linear time
 \rightarrow merge top & bottom into one list

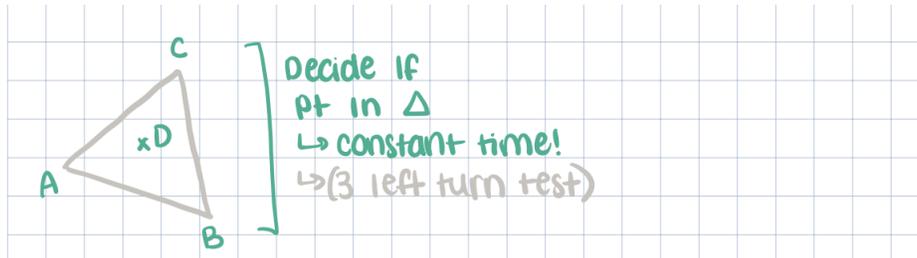
Preprocessing $\Theta(n)$



- if can't make Δ , put on stack
 - pop off as they are connected
 - look @:
 1) current
 2) top
 3) top of stack

Triangulating $\Theta(n)$
 \rightarrow using DCEL

Determining if a point is inside a triangle: $O(1)$

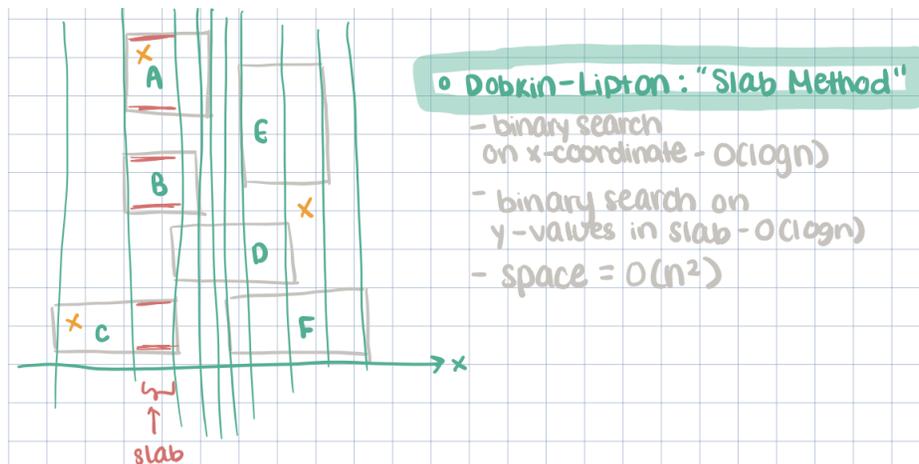


Dobkin-Lipton: Slab Method

Binary Search on X-Coordinate: $O(\log n)$

Binary Search on Y-Coordinate: $O(\log n)$

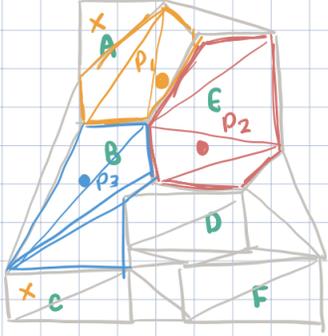
Space: $O(n^2)$



Triangulating

Preprocessing: $O(n \log n)$

Space: $O(n)$



Triangulate Region ~

- preprocessing $\sim O(n \log n)$
- space $\sim O(n)$

◦ make copy of graph w/ each vertex pointing to its copy

◦ place all vertices w/ degree ≤ 11 into candidate set $\rightarrow S$

◦ pick P, remove all neighbors, remove all connected edges, retriangulate w/ P ptrng to each rem Δ

◦ can throw out $\frac{1}{2}$ pts every time

◦ space $\sim S(n) = n + S(\frac{23n}{24}) = \Theta(n)$

while $S \neq \emptyset$

◦ each time pull pt out, 2 fewer Δ s

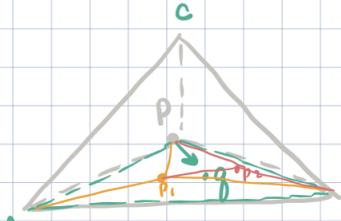
Hierarchical Search

Can use this algorithm to locate a point inside a triangle

Preprocessing: $O(n \log n)$

Query Time: $O(\log n)$

Space: $O(n)$



Hierarchical Search

- preprocessing = $O(n \log n)$
- space = $O(n)$
- query time = $O(\log n)$

↳ reference pt in Δ to divide into 3 parts

↳ continue doing until get to the bottom

Kirkpatrick: $\frac{1}{2}$ of the vertices have degree < 12

Euler's Theorem

