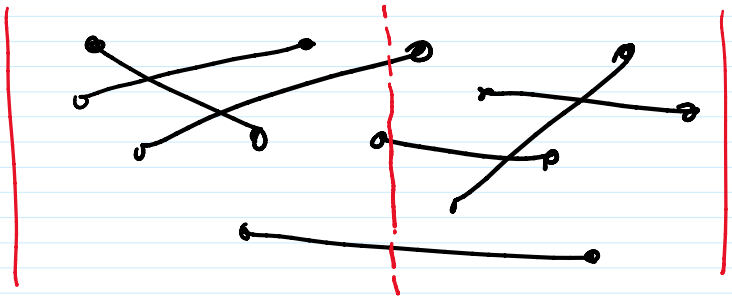
October 31, 2024 3:18 PM

# Line Segment Intersection

Bentley-Ottmann
$O(n \log n + k \log n)$

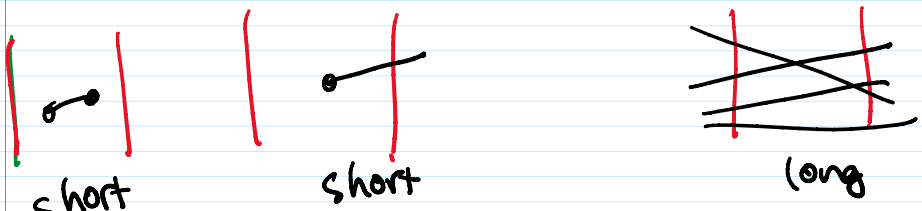
## A Divide-& Conquer Algm (Balaban '95)

(compute intersection but not arrangement)



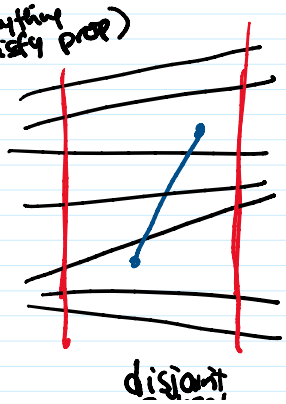
```
intersect(S, σ):    // given slab σ
1. divide σ into σ1, σ2 by median x
2. for i=1,2 {
3.     Si = segs intersecting σi
4.     filter-long(Si, σi)
5.     intersect(Si, σi)
   }
```

**Def** A seg is short if it has an endpt in $\sigma$
long else




**filter-long(S, σ):** (can't insert anything & still satisfy prop)

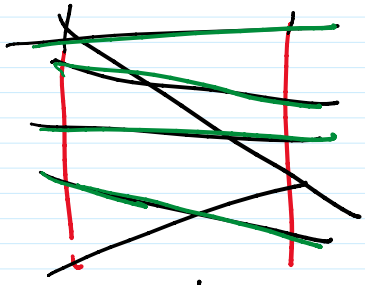
1. find a maximal disjoint subset A of long segs
2. for each $s \in S$
   report all intersections of s with A

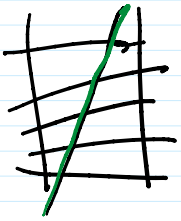


report all intersections of $s$ with $A$ by binary search

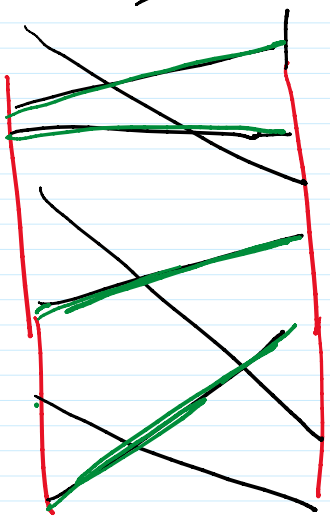
$O(n \log n + k)$

3. remove $A$ from $S$.





disjoint subset



greedy for maximal

Step 1:

$O(n)$ time after sorting

Analysis:

examine recursion tree

at each node $v$, let $n_v$ = # short segs

$\ell_v$ = # long segs

$$\text{cost} = O((n_v + \ell_v) \log n) + \text{output}$$

Observe (i) $\sum_v n_v = O(n \log n)$

(ii) $\sum_v \ell_v = O(k)$

since each long seg can be charged to an intersection pt reported.

$\Rightarrow$ total time $O((n \log n + k) \log n)$

$\Rightarrow$ total time $O((n\log n + k)\log n)$

$= O(n\log^2 n + k\log n)$

**Refinement:**

- cost per node $= O(n_v \log n + l_v)$

  by assuming segs pre-sorted at left wall of $\sigma$

  $\Rightarrow$ total sums to

  $\boxed{O(n\log^2 n + k)}$

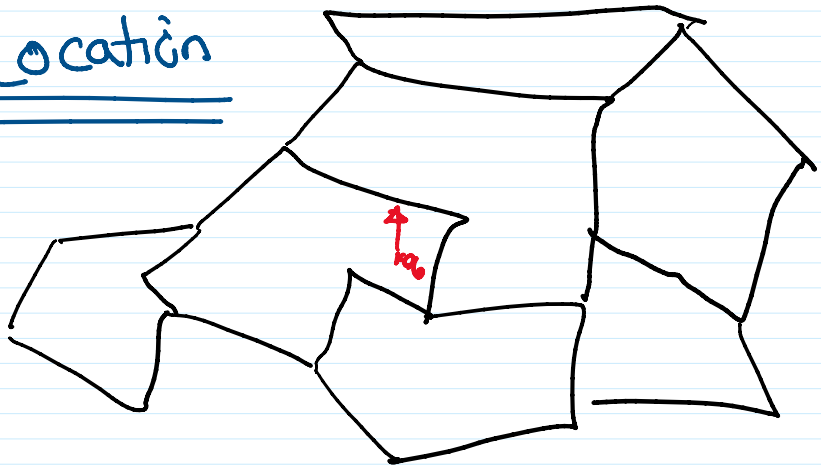
- "fractional cascading"

  $\Rightarrow$ $\boxed{O(n\log n + k)}$

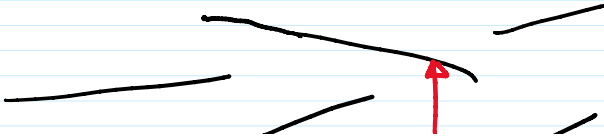
  $O(n)$ space

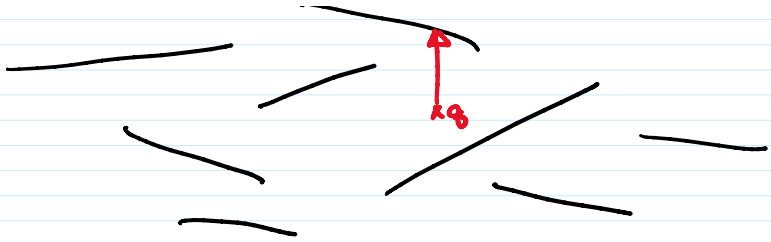
---

## Point Location



**Problem** preprocess a set $S$ of disjoint line segments s.t. given query pt $q$, can find seg immediately above $q$.

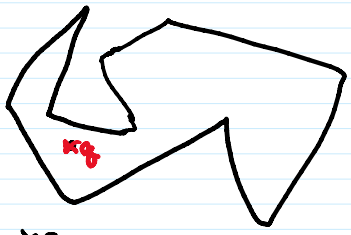






Applns - post office / nearest neighbor queries

- pt in nonconvex polygon?



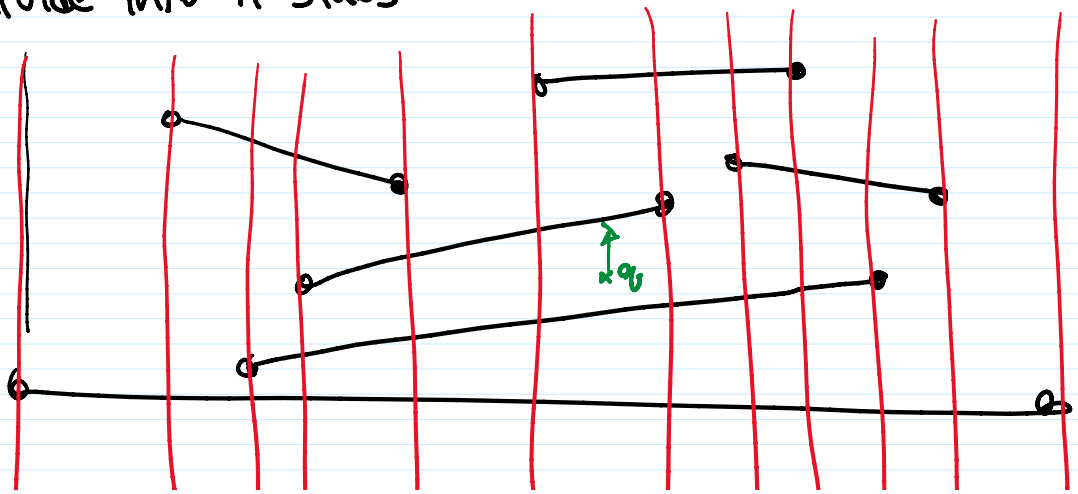

- database, graphics, ...

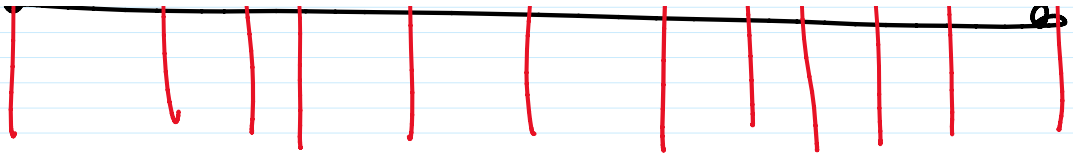
$d=1$: preproc. $P(n) = O(n \log n)$ ( $O(n)$ if sorted)
space $S(n) = O(n)$
query time $Q(n) = O(\log n)$ (binary search)

$d=2$ ?

"~~Brute force~~" Slab Method (Dobkin-Lipton '76)

divide into $n$ slabs



$Q(n) = O(\log n)$ (binary search in x then in y)

$S(n) = O(n^2)$ worst-case

$P(n) = O(n^2 \log n)$

## Divide & Conquer Method ⇒ Segment Trees