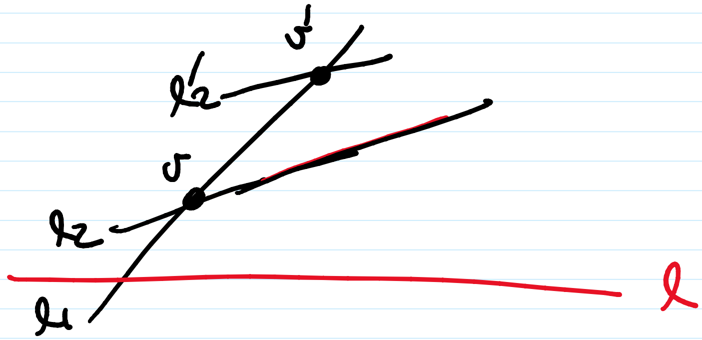
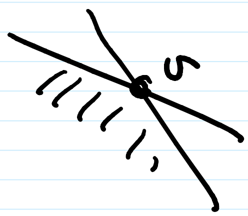
charge v to ℓ_1

Each line charged once



$\Longrightarrow$ $\leq n$ such vertices

Type 3:



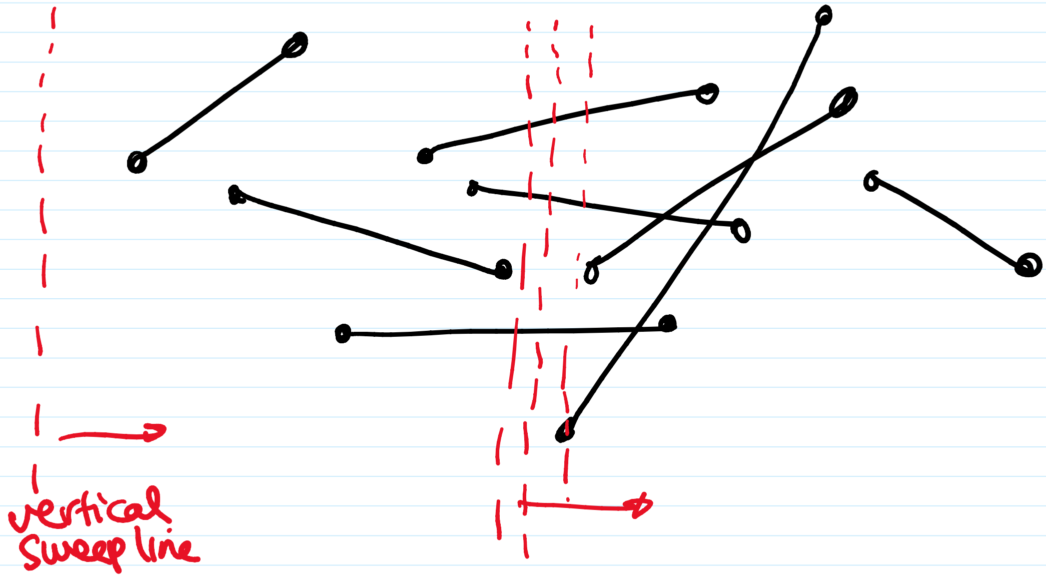
symmetric

$\square$

Rmks: $O(n^d)$ time in $\mathbb{R}^d$ (hyperplane arrangement)

- for line segments: also $O(n^2)$ time
  not output-sensitive
- randomized incremental: $O(n\log n + k)$ expected time
  (Clarkson-Shor '89 / Mulmuley '89)
  (more on this later...)

Sweep Alg'm for Line Segments (Bentley-Ottmann '79)



x-coord as time

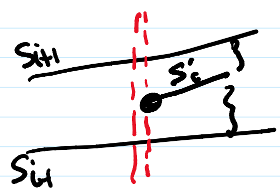
2D problem $\Rightarrow$ dynamic 1D problem

first data structure

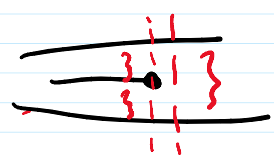
- set T of segments intersecting $x=t$
  sorted by y
- use a balanced search tree
  insert, delete } $O(\log n)$ time
  pred, succ

When does T change?

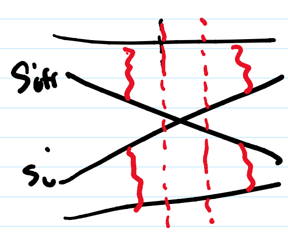
left endpt event

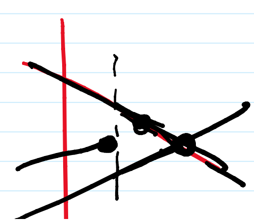


right endpt event

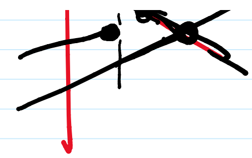


intersection event





Second data structure



Second data structure

- set Q of all endpts & intersection pts of consecutive pairs (s_i, s_{i+1}) right of $x=t$

  sorted in x

- use a heap

  $\Rightarrow$ insert, delete in Q in $O(\log n)$

  min in $O(1)$

Pseudocode:

$Q =$ all endpts, $t = -\infty$

while $Q \neq \phi$ {

  $q =$ Q.delete-min()

  $t = q.x$

  if (q is left endpt of some s_i) {

    T.insert(s_i), $s_{i-1} =$ T.pred(s_i), $s_{i+1} =$ T.succ(s_i)

    Q.insert($s_{i-1} \cap s_i$), Q.insert($s_i \cap s_{i+1}$),

      Q.delete($s_{i-1} \cap s_{i+1}$)

  }

  if (q is right endpt of some s_i) {

    $s_{i-1} =$ T.pred(s_i), $s_{i+1} =$ T.succ(s_i)

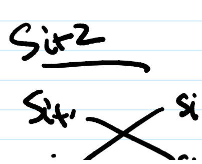
    T.delete(s_i)

    Q.delete($s_{i-1} \cap s_i$), Q.delete($s_i \cap s_{i+1}$),

      Q.insert($s_{i-1} \cap s_{i+1}$)

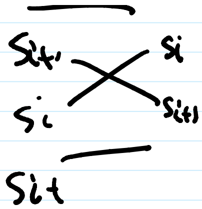
  }

  if (q is an intersection $s_i \cap s_{i+1}$) {



    $s_{i-1} =$ T.pred(s_i), $s_{i+2} =$ T.succ(s_{i+1})

    T.swap(s_i, s_{i+1})



$s_{i-1} = T.\mathrm{pred}(s_i)$, $s_{i+2} = T.\mathrm{succ}(s_{i+1})$

$T.\mathrm{swap}(s_i, s_{i+1})$

$Q.\mathrm{delete}(s_{i-1} \cap s_i)$, $Q.\mathrm{delete}(s_{i+1} \cap s_{i+2})$

$Q.\mathrm{insert}(s_{i-1} \cap s_{i+1})$, $Q.\mathrm{insert}(s_i \cap s_{i+2})$

output q

}

}

Analysis: # iterations/events $O(n+k)$

each iter $O(\log n)$

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

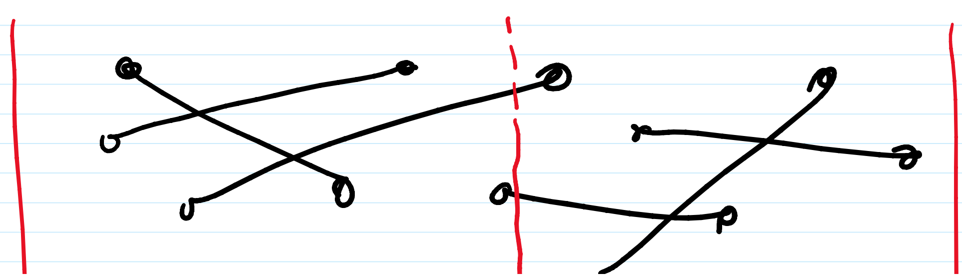
$\boxed{O(n)}$ space

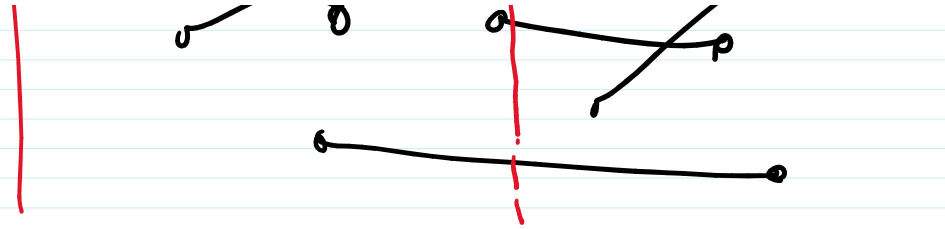
Rmks:

- detection in $O(n \log n)$ time
- can compute arrangement & also trapezoidal decomposition
- Chazelle-Edelsbrunner '88: $O(n \log n + k)$ time by a complicated sweep $O(n+k)$ space

A Divide-& Conquer Alg'm (Balaban '95)

(compute intersection but not arrangement)

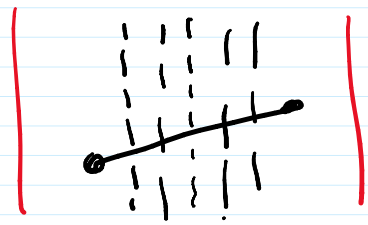




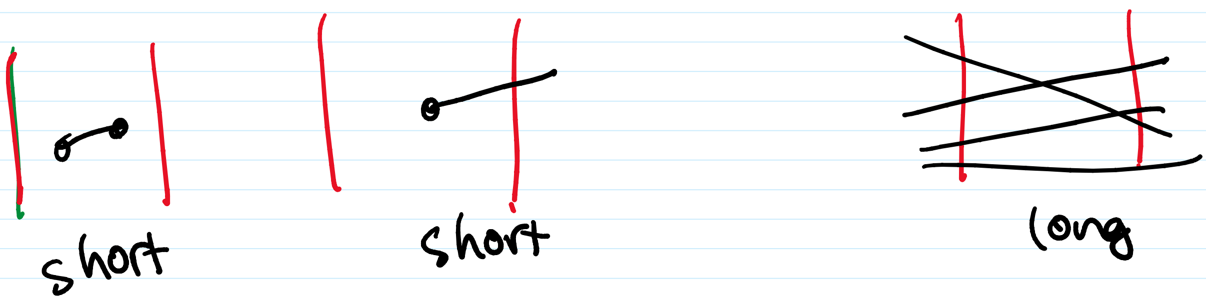
intersect(S, σ):   // given slab σ

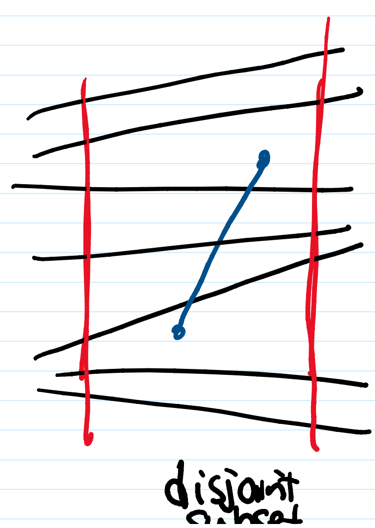
1. divide σ into σ_1, σ_2 by median x
2. for $i = 1,2$ {
3. S_i = segs intersecting σ_i
4.   filter-long(S_i, σ_i)
5.   intersect(S_i, σ_i)

   }



Def A seg is <u>short</u> if it has an endpt in σ
<u>long</u> else





filter-long(S, σ):

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

disjoint
subset

3. remove A from S.