Given a set $S$ of $n$ line segments:

- Do any two intersect?
- Which pairs intersect?
- How many pairs intersect?
- Find/count intersection points
- Subdivide $S$ at all intersect points.

Arrangement of $S$: $O(n^2)$ time brute force.

Arrangement is planar straight-line graph (PSLG):
Vertices = endpoints & intersection points
Edges = maximal subsegments.

Examples:
- Given $Z$ polygons, compute union/intersection/convex hull.
- Given a single polygon, is it simple?

Generalizes to lines, boxes, circles, curves, as in 3D.
Two segments

Solve for lines
Solve for point
Check if pt is on segments

orient(pqr) = \text{sign} \ det \begin{bmatrix} 1 & p_x & p_y \\ 1 & q_x & q_y \\ 1 & r_x & r_y \end{bmatrix} = \begin{cases} 0 & \text{ccw} \\ >0 & \text{f1st} \\ <0 & \text{cw} \end{cases}

pq \cap rs \neq \emptyset
iff
orient(pqr) \neq orient(pqs)
and
orient(prs) \neq orient(ars)

For now assume general position
- no equal coordinates
- no 3 endpoints collinear

Detecting intersections
Use sweep line algorithm
Sweep line maintains sorted seq. of segments if it intersects
Left endpoint: insertion
Right endpoint: deletion
Crossings involve adjacent indices

"Sweep line structure" is balanced BST
Implicity use y-coords of intersection with l
as "keys" but computed on the fly

Left end: insert $O(\log n)$ time
check new seg and successor $\gtrsim O(\log n)$ time
check new seg and predecessor $\lesssim O(\log n)$ time

Right end: delete $O(\log n)$ time
check pred + succ of old segment $O(\log n)$ time

Sort endpoints →
For $i = 1$ to $2n$
if $p_{[i]}$ is left
$\gtrsim O(\log n)$
else
$\lesssim O(\log n)$

$\gtrsim O(n\log n)$
$\lesssim O(n)$
$\times O(\log n)$

$O(n\log n)$ time

Count/list all intersections
Swap segments when they intersect

At most 2 segs intersect at one pt
Maintain a priority queue of events "event queue"

Init: insert all 2n endpoints into PQ

When we detect intersection

Add x-coord of that intersection to PQ

Init PQ with endpoints

while PQ is not empty

x = next event from PQ

if x is left end

O(log n)\

if new seg + succ intersect

add x-coord of int to PQ

if new seg + pred intersect

add PQ

update BST

if x right

O(log n)\

if x intersection

O(log n)\

swap in BST

check 2 new pairs

Overall running time = \( O(\log n) \times \# \text{ events} \)

= \( O(nk \log n) \)

\( 2n + k \)

Later: \( O(n \log n + k) \) time — different method
Counting intersections: $O(n^{4/3} + 2)$

Very different method.