Planar Point Location:
Given a subdivision of the plane (cell complex), build a data structure so that for any query pt, can find the cell containing it.

Vertical Ray Shooting:
Given a set of disjoint line segments in the plane, build a data structure s.t. given any query pt $q$, can report the segment immediately below.

Ray Shooting $\Rightarrow$ Point Location
Label each segment with region just above
Data structure for vertical ray shooting:

Approach: Build trapezoidal map + ray shooting structure simultaneously

\[ S = \{s_1, \ldots, s_n\} \quad \text{Randomly permuted} \rightarrow \mathcal{T}(S) \]
\[ S_i = \{s_i, \ldots, s_i\} \quad \rightarrow \text{Partial map} \quad \mathcal{T}(S_i) = \mathcal{T}_i \]

Recall: In expectation, each insertion results in \(O(1)\) changes to structure.

Overview:
- Rooted binary tree with shared subtrees (a rooted DAG)
- Each leaf corresponds to a trapezoid
- Each trapezoid occurs exactly once as leaf
- Internal nodes - two types

\(x\)-Node:
- Labeled with an end pt \(p\)
**Example:**

![Diagram of a tree structure with labeled segments and nodes.](image)

**Query processing:**

![Diagram showing query processing steps.](image)
Incremental Construction:

- As segments are added: \( s_1, s_2, \ldots, s_i \), we build structure for \( \mathcal{T}(s_1), \mathcal{T}(s_2), \ldots \mathcal{T}(s_i) \)

- Update process:
  - Each added segment causes some trapezoids to go away + others created
  - We replace old leaves with new structures
  - By sharing, only one leaf per trapezoid

1: Single end pt in trapezoid (left or right):

(Right end pt is symmetrical)
2: Two segment endpoints in same trapezoid

3: No segment endpoint in trapezoid
Example:

Analysis:

Will show if seqs are inserted in random order, expected space is $O(n) +$ expected search time for any fixed query pt is $O(\log n)$.
Thm: The expected case space is \( O(n) \)

Proof: Last lecture we showed that expected no. of changes is \( O(1) \) per seg \( \Rightarrow \) total changes \( O(n) \)

Number of new nodes \( \sim \) number of changes

\( \Rightarrow \) final expected size is \( O(n) \)

Thm: Given a fixed query pt \( q \in \mathbb{R}^2 \), the expected search depth for \( q \) is \( O(\log n) \)

Huh? Does this imply that depth of search tree is \( O(\log n) \) in expectation?

No - But see our text for a proof of this.

Proof:

- Let \( q \) be any fixed query pt.

- Let \( \Delta_i(q) \) be the trapezoid containing \( q \) after the insertion of \( s_i \) (\( 1 \leq i \leq n \))

  - Note: Sometimes \( \Delta_i(q) = \Delta_{i-1}(q) \) (\( s_i \) had no impact)

  - What if \( \Delta_i(q) \neq \Delta_{i-1}(q) \)?
- For $1 \leq i \leq n$, let $X_i(q) = \begin{cases} 1 & \text{if } \Delta_i(q) \neq \Delta_{i-1}(q) \\ 0 & \text{o.w.} \end{cases}$

- If $X_i(q) = 1$, \( \text{depth}(\Delta_i) \leq 3 + \text{depth}(\Delta_{i-1}) \)

Let $D(q)$ the expected depth of $q$'s trapezoid in the final structure.

$$D(q) \leq 3 \sum_{i=1}^{n} \mathbb{E}(X_i(q))$$

$$= 3 \sum_{i=1}^{n} \text{Prob}(\Delta_i(q) \neq \Delta_{i-1}(q))$$
- We assert that \( \text{Prob}(\Delta_i(q) \neq \Delta_{i-1}(q)) \leq \frac{4}{i} \)

- Backwards analysis:
  - Each of the existing \( i \) segs is equally likely to be last (prob = \( \frac{1}{i} \))
  - \( \Delta_i(q) \neq \Delta_{i-1}(q) \) iff last segment is one of the 4 segments incident to \( \Delta_i(q) \)

\[ \Rightarrow \text{Prob}(\Delta_i(q) \neq \Delta_{i-1}(q)) \leq \frac{4}{i} \]

- Substituting: Expected depth of \( q \)'s trapezoid

\[
D(q) \leq \sum_{i=1}^{n} \text{E}(X_i(q)) = 3 \sum_{i=1}^{n} \text{Prob}(\Delta_i \neq \Delta_{i-1})
\]

\[ \leq 3 \sum_{i=1}^{n} \frac{4}{i} = 12 \sum_{i=1}^{n} \frac{1}{i} \text{ (Harmonic series)}
\]

\[ \approx 12 \ln n = \Theta(\log n) \]
Summary:

- Last time we showed that randomized incremental alg. took $O(1)$ time in expectation per segment, ignoring time to locate left end pt.

- Today, we presented a data structure with query time $O(\log n)$ for pt location.

$\implies$ Total expected construction time is:

$$T(n) = \sum_{i=1}^{n} ((\log i) + 1)$$

(update structure)

(locate left end pt)

$= O(n \log n)$

- Space + Query time are in expectation.
- Can we guarantee them?
  - Yes: Just rebuild if things go wrong (increases expected construct time slightly, but still $O(n \log n)$.)

(see text for details)
Line segment intersection (Revisited):

- Can extend trap. maps to intersecting segs.

- Randomized construction can be easily generalized.

Expected time: $O(n \log n + m)$

where $m = \# \text{ of intersections}$

This beats plane sweep! $O((n+m) \log n)$