Randomized Incremental Algorithm for trapezoidal decomposition of array of $n$ segments in $\mathbb{R}^2$

**Problem 1.** Given a set of line segments in the plane, subdivide the plane into a collection of trapezoids, formed by shooting a bullet to the left and right of each vertex and each intersection point until it hits the first object. Design an algorithm to find out all crossing points such that running time is better than that of naive solution $O(n^2)$.

**Deterministic Algorithm:**

- Easy-sweep costs $O((n + k) \log n)$. It is simple.
- $O(k + n \log n)$, but it is very complex.

**Our Goal:**

- simple randomized incremental algorithm and give $O(k + n \log n)$ with high probability.

The basic idea of the algorithm is described below.

**Randomized Incremental Algorithm:**

1. Randomly permute the set of segments: $S = \{s_1, s_2, \ldots, s_n\}$. Let $N^i$ denote the first $i$ segments, e.g. $N^i = \{s_1, \ldots, s_i\}$. In general, $H(N^i)$ will denote the trapezoidal decomposition of the first $i$ segments $N^i$.

2. Initialize the structure to a trivial "empty" trapezoidal decomposition (e.g. a large empty enclosing box), $H(N^0)$. 
(3) one by one add the segments in random order. For each segment, do the following:

(a) Locate the trapezoid of $H(N^{i-1})$ containing the left endpoint of the segment.
   (There are a couple of ways to do this. More on this later.)

(b) Trace the segment from one trapezoid to the next. At the endpoints of the segment, split the current trapezoid by adding a vertical wall. For each trapezoid determine the point of entry and point of exit of the segment. If the segment crosses the top or bottom then create new intersection points, and split the trapezoid by adding a vertical wall.

(c) After tracing the segment, determine the vertical walls of $H(N^{i-1})$ that were stabbed by the segment. For each such wall, trim it back to the portion containing its supporting vertex. This results in the merger of two adjacent trapezoids.

Figure 1: Randomized incremental trapezoidal decomposition

We assume that we can (1) trace a segment through a face of the decomposition, (2) split a face and (3) merge two adjacent faces in time proportional to the complexity of the face,
that is, the number of vertices on the face. We let \( f \) denote a trapezoidal face of the decomposition, and let \( c(f) \) denote the "complexity" of this face. Observe that the running time of the above algorithm is \( O(c(f)) \) for each face that is traced by this procedure, because there can only be a constant number (at most 2) split performed for each trapezoid, and a constant number (at most 2) merges for each trapezoid.

The other question is how to locate the left-endpoint from which to start the tracing. We do this using a simple bucketing strategy. For each of the \( n \) left-endpoints, we store which trapezoid of the current subdivision contains this point, and with each trapezoid \( f \) we associate a list \( L(f) \), which contains the segments whose left-endpoint lies within this face. When we wish to trace a segment, we can determine the trapezoid containing this segment in constant time. When a segment intersects a trapezoid \( f \), it may split a trapezoid \( f \) into a constant number of new trapezoids. We can walk through the list \( L(f) \) and determine which of the new trapezoids contains a given point in constant time each. We form new left endpoint lists for each of the new faces, \( L(f_1) \), \( L(f_2) \), etc. When we merge two trapezoids, we simply concatenate their lists. Observe that both of these operations can be performed in time \( O(l(f)) \), where \( l(f) = |L(f)| \).

![Diagram of splitting a face](image)

\[
c(f) = 9 \quad l(f) = 5
c(f_1) = 7 \quad l(f_1) = 2
\]

\[c(f_2) = 5 \quad l(f_2) = 2
\]

\[c(f_3) = 4 \quad l(f_3) = 1
\]

**Figure 2:** Splitting a face.

**Lemma 2.** Consider the insertion of segment into \( H(N^{i-1}) \) which intersects faces \( F = \)
\{f_1, f_2, \ldots, f_k\} in the decomposition. The time to insert this segment is

\[ O\left(\sum_{f \in F} (c(f) + l(f))\right). \]

Analyzing the expected case running time of this algorithm is quite a tricky task. We want to show that, no matter what \( n \) segments are given initially, if we randomize of all possible insertion orders, the total expected time to build the decomposition is \( O(k + n \log n) \). In particular, we seem to need to be able to determine the expected value of \( \sum f(c(f) + l(f)) \) over all segments that might be inserted next.

This task would be quite daunting, if it were not for a very clever analysis trick, called backward analysis. Here is the idea: we imagine that we are running time algorithm backwards, deleting segments one at a time. Observe that if all insertion orders are equally likely, then the last segment to be deleted in the reversed algorithm is equally likely to be any one of the segments that already exists in the decomposition \( H(N^i) \). Since every segment of \( N^i \) is equally likely to be deleted, to determine the expected time for the \( i \)-th stage, it suffices to average over all possible segments \( i \) to be deleted, and for each determine the complexity if this were the segment chosen.

When we delete some segment from the \( H(N^i) \) observe that the only trapezoids that would be affected from its deletion consist of the trapezoids of \( H(N^i) \) that are adjacent to this segment.

**Lemma 3.** For each \( j, 1 \leq j \leq i \), let \( F_j \) denote the faces that the \( j \)-th segment intersects in \( H(N^i) \). The expecte time for the last stage in the construction of \( H(N^i) \) is on the order of

\[ E(i) = \frac{1}{i} \sum_{j=1}^{i} \sum_{f \in F_j} (c(f) + l(f)). \]

We seem to be no closer to our goal, since there does not appear to be an easy way to analyze the sums of complexities of all the trapezoids that a given segment intersects. However, the crucial trick is not to count segment by segment, but to count trapezoid by trapezoid. Assuming that the segments are in general position, observe the following important fact.

**Lemma 4.** (Bounded degree property) Each trapezoid in a trapezoidal decomposition is adjacent to at most 4 segments. (Namely the segments immediately above and below, and the segments which touch the left and right walls of the trapezoid either because of an endpoint or an intersection point).
Figure 3: Affected trapezoids.

Thus if we take the complexity of each trapezoid, and multiply by 4, we will have an upper bound on the sum of complexities of all the trapezoids that are intersected by all segments.

\[
E(i) \leq \frac{1}{i} \sum_{f \in H(N^i)} 4(c(f) + l(f)) \\
\leq \frac{4}{i} \left( \sum_{f \in H(N^i)} c(f) + \sum_{f \in H(N^i)} l(f) \right).
\]

We know the value of \(\sum_f c(f)\) is the sum of all the edges summed over all the faces in the decomposition. However, this counts every edge in the decomposition twice. Since the decomposition is a planar graph, this is proportional to the number of vertices in the decomposition currently. The value of \(\sum_f l(f)\) is just equal to \(n - i\), since it includes all the left endpoints of segments that have yet to be added. If we let \(k_i\) denote the number of intersection points at the current stage of the decomposition, the number of vertices in
$H(N^i)$ is just $2i + k_i$, so the above formula simplifies to

\[
E(i) \leq \frac{4}{i} (2i + k_i + n - i) \\
\leq \frac{c(n + k_i)}{i}
\]

for some constant $c$. The interesting thing at this point is that, although our analysis was conditional on the structure of $H(N^i)$, the resulting bound is almost entirely independent of this structure.

At this point we can see where the analysis is going. If we ignore the $k_i$ term in the above, we see that $E(i) \leq cn/i$. To get the total time, we sum these up, getting at total expected time of

\[
TE(n) = \sum_{i=1}^{n} E(i) \\
\leq \sum_{i=1}^{n} \frac{cn}{i} \\
= cn \sum_{i=1}^{n} \frac{1}{n} \\
\approx cn \ln n \in O(n \log n)
\]

The last line uses the well known fact that the Harmonic series, $\sum_{1 \leq i \leq n} 1/i$ tends to $\ln n$.

We need to get a handle on what $k_i$ is expected to be. Obviously the value of $k_i$ should eventually approach $k$, the total number of intersections as $i$ approaches $n$. The interesting fact is that this quantity approaches $k$ quadratically, not linearly.

**Lemma 5.** For fixed $i \geq 0$, the expected value of $k_i$, assuming that $N^i$ is a random sample of $N$ of size $i$, is $O(\frac{ki^2}{n^2})$.

**Proof.** For each intersection point $v$ between two segments, $s_1$ and $s_2$, let $I_v$ denote the random variable that this 1 if this intersection is part of the decomposition $H(N^i)$ and 0 otherwise. Observe that the expected value of $k_i$ is $\sum_v I_v$. However, $v$ occurs within $I_v$ if and only if both $s_1$ and $s_2$ occur within the first $i$ randomly selected segments. The probability that each one has been selected alone is $i/n$. The probability that both have been selected is roughly $i^2/n^2$ (for $i$ and $n$ large). Thus the expected value of $k_i$ is $k$ times this quantity, or $O(\frac{ki^2}{n^2})$. \qed

6
To complete the analysis, we make use of another basic fact from summations. $\sum_{1 \leq i \leq n} i = O(n^2)$.

\[
TE(n) = \sum_{i=1}^{n} E(i) \\
\leq \sum_{i=1}^{n} \frac{c(n + k_i)}{i} \\
= \sum_{i=1}^{n} \frac{cn}{i} + \sum_{i=1}^{n} \frac{ck_i}{i} \\
= cn \sum_{i=1}^{n} \frac{1}{i} + c \sum_{i=1}^{n} \frac{ki^2}{n^2i} \\
\approx cn \ln n + \frac{ck}{n^2} \sum_{i=1}^{n} i \\
\approx cn \ln n + \frac{ck}{2} \\
\in O(n \log n + k).
\]