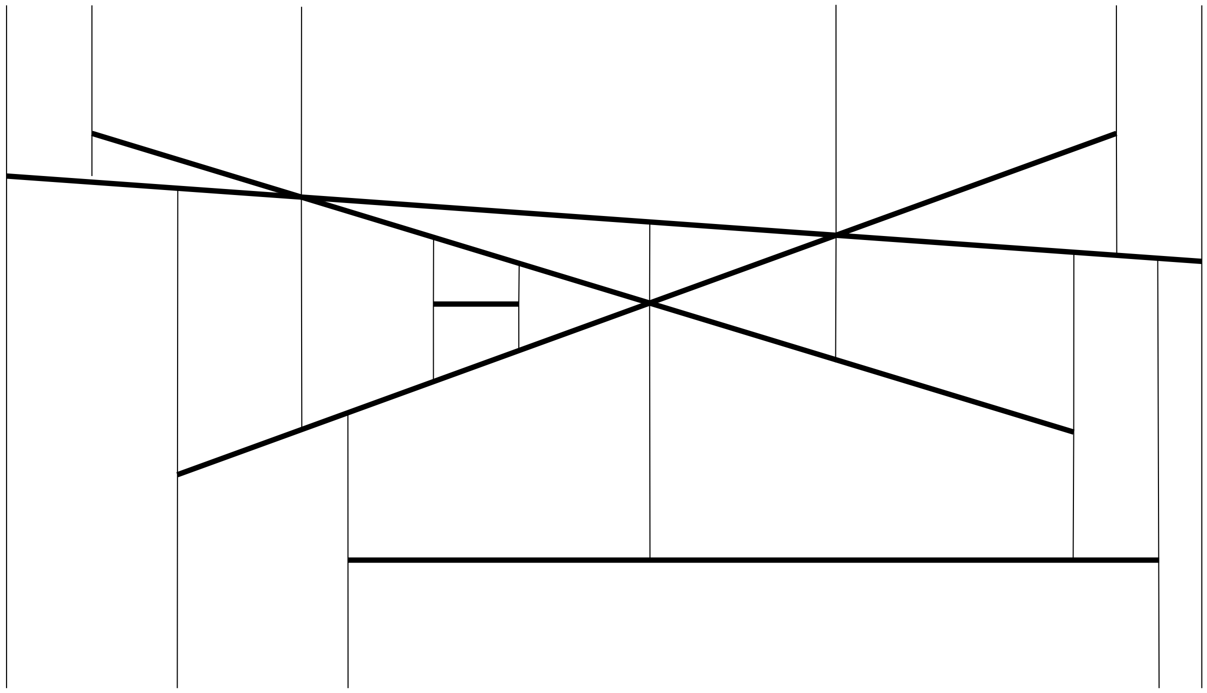
2. Trapezoidal decomposition

$\boldsymbol{N}$: a set of $\boldsymbol{n}$ line segments (possibly unbounded)

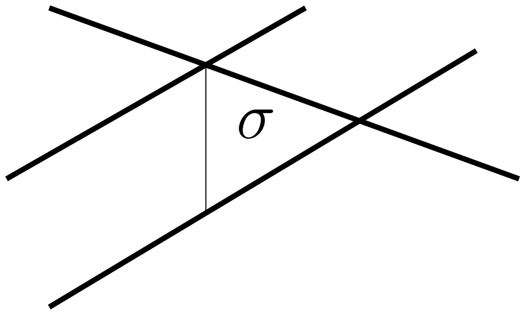
Vertical Trapezoidal Decomposition $\boldsymbol{H(N)}$ of N

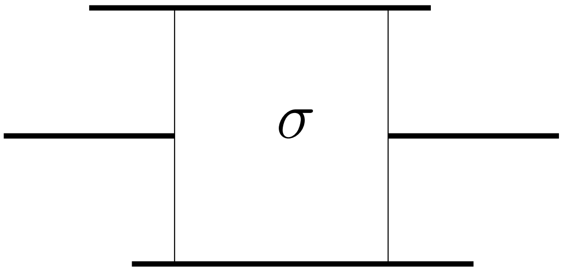
- Pass a vertical attachment through every endpoint or point of intersection
- Each vertical attachement exteneds upwards and downwards until it hit another segment or if no such segment exist, it extends to infinity



Properties of $\boldsymbol{H(N)}$

- Each cell is called a **trapezoid** and consists of at most 4 edges (either triangle or quadrilateral)
- Each cell is defined by at most four line segments





The Sorting Problem:

Find the vertical trapezoidal decomposition $H(N)$

The Search Problem:

Associate a search structure $\widetilde{H}(N)$ with $H(N)$, so that for a give query point q, locating which trapezoid of $H(N)$ it belongs to is efficient

Randomized Incremental Construction:

- Generate a random seqeuence $S_1, S_2, \ldots, S_n$ of N
- Construction $H(N)$ by iteratively adding $S_1, S_2, \ldots, S_n$, i.e., computing $H(N^0), H(N^1), \ldots, H(N^n)$ iteratively, where $N^0 = \emptyset$ and $N^i = \{S_j \mid 1 \leq j \leq i\}$

2.1 Conflict List

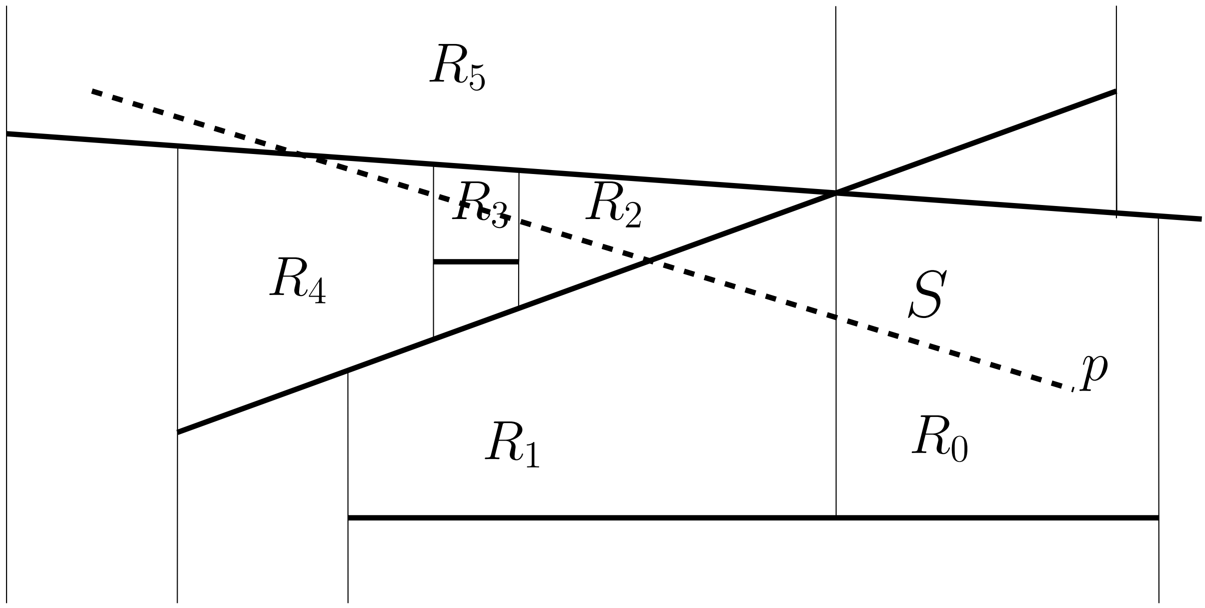
Assume $\boldsymbol{H(N^i)}$ are avaiable

Conflict relations are defined between trapezoids of $H(N^i)$ and endpoints of line segments of $N \setminus N^i$

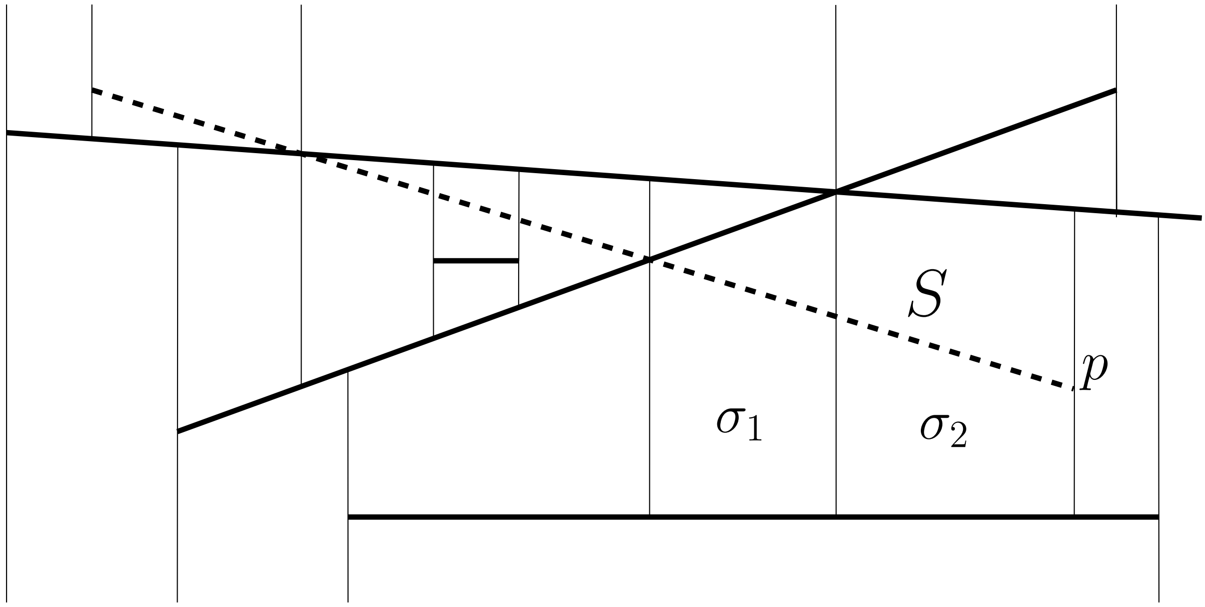
- For each trapezoid of $H(N^i)$, store the endpoints of line segments of $N \setminus N^i$ located in it
- For each endpoint of $N \setminus N^i$, store the trapzezoid of $H(N^i)$ to which it belongs

Adding $S = S^{i+1}$ to obtain $H(N^{i+1})$

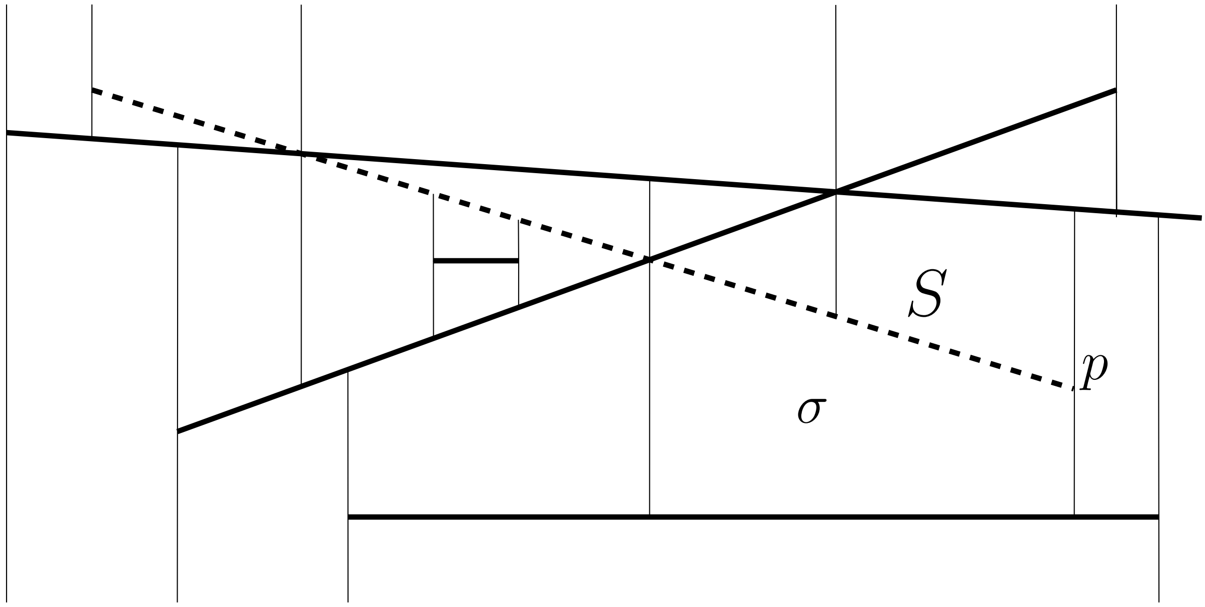
1. Find out the trapezoid including an endpoint p of S^{i+1}
2. Travel from p to trace out all the trapezoid of $H(N^i)$ intersecting S
3. Spilt all the traced trapezoids by S
4. Combine adjacent trapezoids whose upper and lower edges are adjacent to the same segments



Before Inserting S



Split

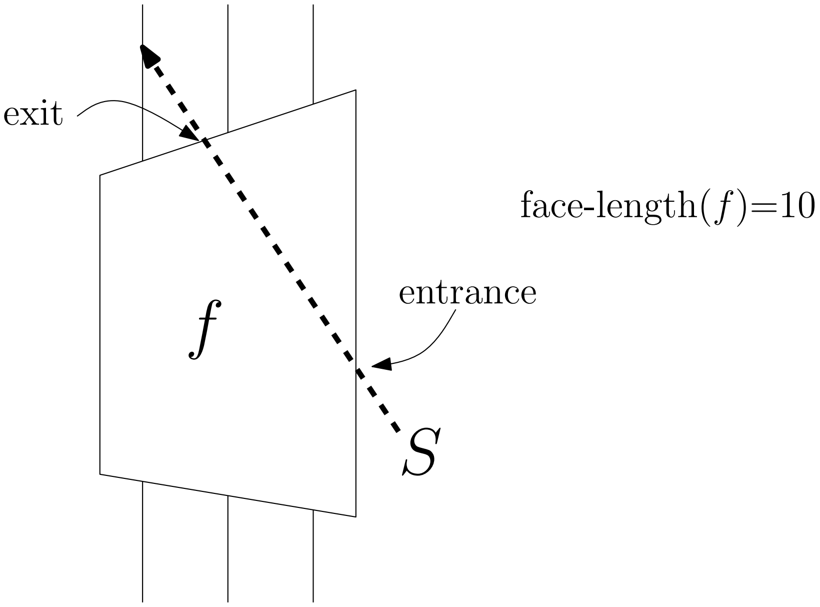


Merge (e.g., merging σ_1 and σ_2 into σ)

How to trace $R_0, R_1, \ldots, R_j$ of $H(N^i)$ intersecting S

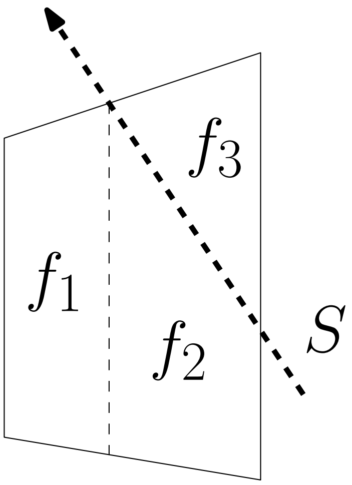
Let f be the current traced trapezoid during the travel

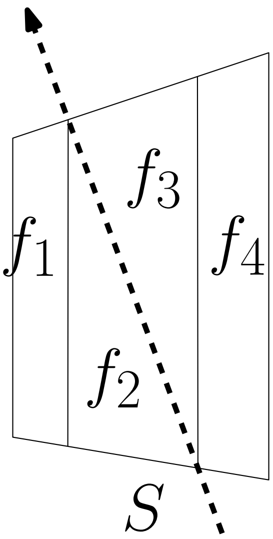
- Traverse the boundary of δ to find the exit point
- Time proportional to face-length(f), which is number of vertices of f in $H(N^i)$

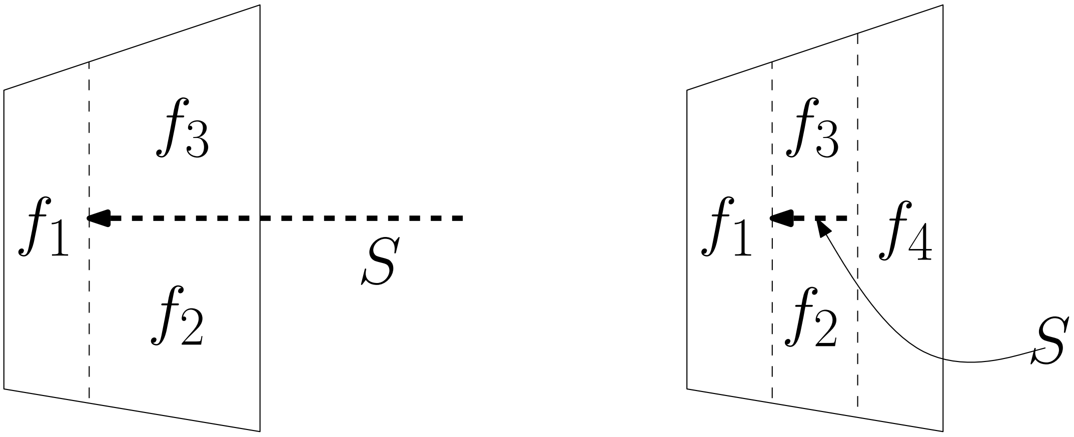


How to split an trapezoid f

- If S intersect the upper or lower side of f, raise a vertical attachment from the intersection within f
- If an endpoint of S is inside f, raise a vertical attachement from the endpoint within f
- At most four new trapezoid

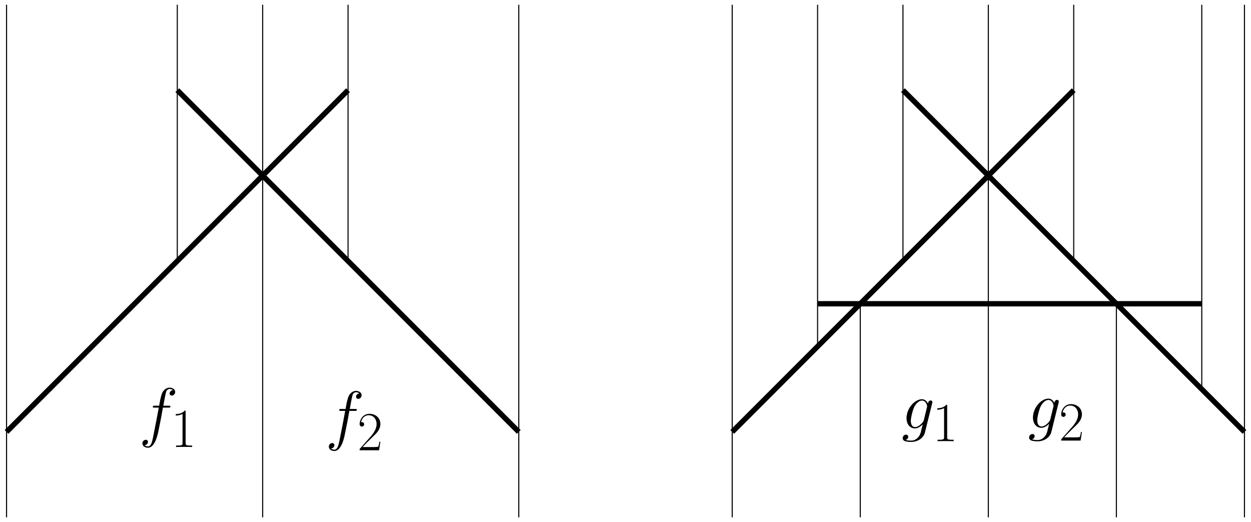






Why and How to Merge

- Two new trapezoids from difference trapezoids in $H(N^i)$ may belong to the same trapezoid in $H(N^{i+1})$
- If two adajcent new trapezoids share the same top and bottom segments, merging them takes $O(1)$ time



g_1 and g_2 belong to f_1 and f_2, respectively, and will be merged

Proposition 2.1
Once we know the trapezoid in $H(N^i)$ containing one endpoint of $S = S^{i+1}$, $H(N^i)$ can be updated to $H(N^{i+1})$ in time proportional to $\sum_f$ face-length(f), where f ranges over all trapezoids in $H(N^i)$ intersecting S.

How to find the starting trapezoid

- Conflict Lists
- $O(1)$ time by the "edge" from an endpoint of S to the conflicted trapezoid

How to update conflict list
For a trapezoid f, $L(f)$ is endpoints of $N \setminus N^i$ in f, and $l(f)$ is $|L(f)|$

- **Split:** If f is split into $f_1, \ldots, f_i$, $i \leq 4$, for each point $p \in L(f)$, decide f_i which p belongs to in total $O(l(f))$ time
- textbfMerge: $O(1)$ time

Proposition 2.2
The cost of updating conflict lists if $O(\sum_f l(f))$, where f ranges over all trapezoids in $H(N^i)$ intersecting S and $l(f)$ denotes the conflict size of f.

Backward Analysis for Inserting S

Originally: adding S into $H(N^i)$

$$O(\textstyle\sum_f \text{face-length}(f) + l(f))$$

where f ranges over all trapezoids in $H(N^i)$ intersecting S

$$=$$

Now: removing S from $H(N^{i+1})$

$$O(\textstyle\sum_g \text{face-length}(g) + l(g))$$

where g ranges over all trapezoids in $H(N^{i+1})$ adjacent to S

Since $S_1, S_2, \ldots, S_n$ is a randon sequence of N,
each line segment in N^{i+1} is equally likely to be S.

Expected cost is proportional to

$$\frac{1}{i+1} \sum_{S \in N^{i+1}} \sum_g \text{face-length}(g) + l(g)$$

where g ranges over all trapezoids in $H(N^{i+1})$ adajcent to S

It equals to $\frac{n-i+|H(N^{i+1})|}{i+1} = O(\frac{n+k_{i+1}}{i+1})$

where g denotes the number of intersection among the segments in N^{i+1} and $|H(N^{i+1})|$ denotes the total size of $H(N^{i+1})$

because

- Each trapezoid in $H(N^{i+1})$ is adjacent to at most four segments in N^{i+1},
  $\rightarrow \sum_{S \in N^{i+1}} \sum_g \text{face-length}(g) \leq 4|H(N^{i+1})|$
- Total conflicts $\sum_{S \in N^{i+1}} \sum_g l(g)$ is $2(n-i)$
- $|H(N^{i+1})| = O(i+1+k_{i+1})$

Lemma 2.1:
Fix $j \geq 0$, the expected value of k_j, assuming that N^j is a random sample of N of size j, is $O(kj^2/n^2)$

proof is an exercise

Theorem 2.1
A trapezoidal decomposition formed by n segments in the plane can be constructed in $O(kn \log n)$ expected time. Here k denotes the total number of intersections among the n segments

$$E[\sum_{i=0}^{n-1} O(\tfrac{n+k_{i+1}}{i+1})] = \sum_{i=0}^{n-1} E[O(\tfrac{n+k_{i+1}}{i+1})]$$

$$= \sum_{i=0}^{n-1} O(\tfrac{n+ki^2/n^2}{i+1})] = (\sum_{i=0}^{n-1} \tfrac{n}{i+1}) + (\sum_{i=0}^{n-1} ki^2/n^2)$$

$$= \boldsymbol{O(n \log n + k)}$$

Two questions for this randomized incremental construction based on conflict lists

- How about search structure: locate a query point in a trapezoid of $H(N)$
- Not a on-line algorithm because the conflict lists depend on $N \setminus N^i$

2.2 History Graph

On-Line Algorithm and Search Structure

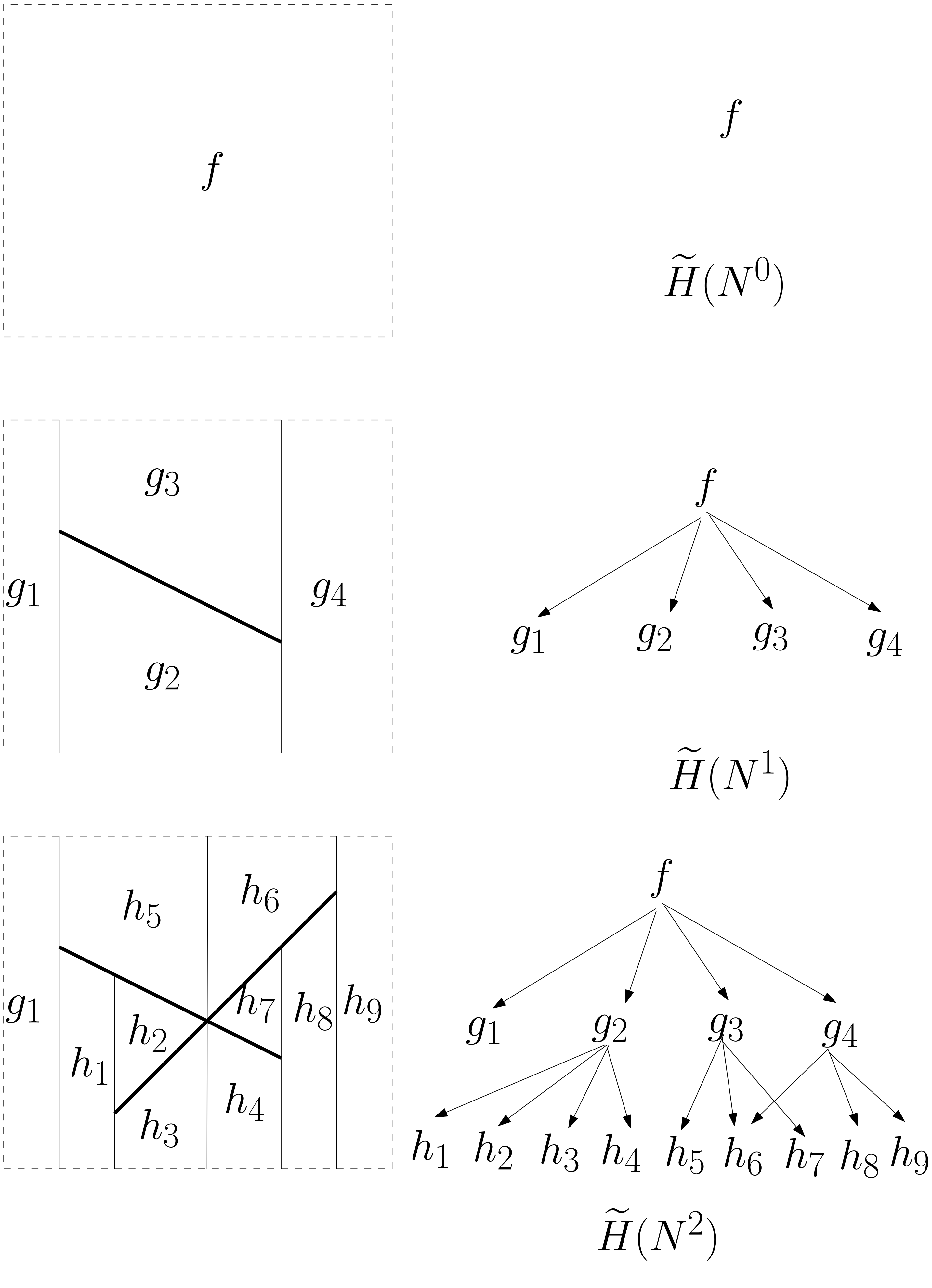
- Recall Random Binary Tree of Quick-Sort
- Killer and Creator
  - All trapezoids in $H(N^i) \setminus H(N^{i+1})$, S^{i+1} is their killer
  - All trapezoids in $H(N^{i+1}) \setminus H(N^i)$, S^{i+1} is their creator

history(i) ($= \widetilde{H}(N^i)$) is a directed graph $G(V, E)$

- V: all trapezoids appeared in $H(N^0), H(N^1), \ldots, H(N^i)$
- E: an arc connectes u to v if
  - The killer of v is the creator of u,
    i.e., the insertion of S kills u and creates v.
  - v and u intersect each other
  - u is called a parent of v, and v is called a child of u.

Properties of history(i) ($= \widetilde{H}(N^i)$)

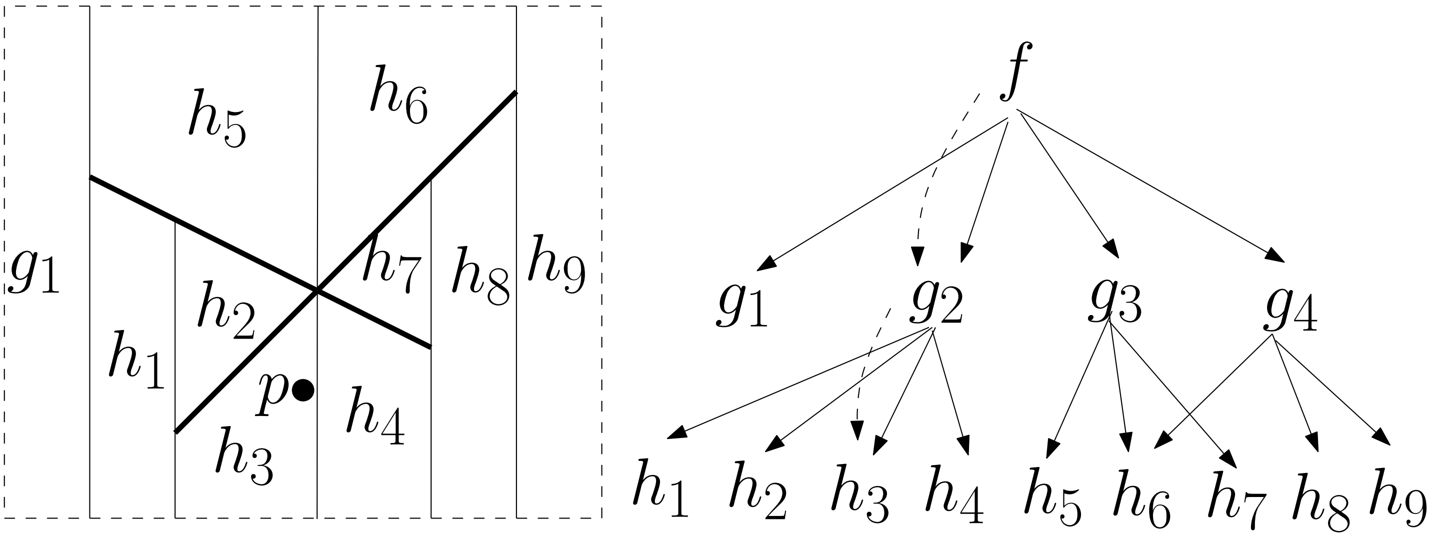
- Its leaves form $H(N^i)$
- $H(N^0)$ is the only vertex without in-going edges and called the root
- It is an acyclic graph
- Each node has at most 4 out-going edges
- If a point p is contained in a trapezoid v, there is a path from the root to v along which each trapezod contains p


f
f
$\widetilde{H}(N^0)$
g_3
g_1
g_4
g_2
f
g_1
g_2
g_3
g_4
$\widetilde{H}(N^1)$
h_5
h_6
g_1
h_7
h_8
h_9
h_2
h_1
h_4
h_3
f
g_1
g_2
g_3
g_4
h_1
h_2
h_3
h_4
h_5
h_6
h_7
h_8
h_9
$\widetilde{H}(N^2)$

Adding S^{i+1} into $H(N^i)$ through $\widetilde{H}(N^i)$

1. Locating an endpoint p of S^{i+1} by $\widetilde{H}(N^i)$

- Starting from the root until a leaf is reached, check which child contains p and search the child



$\widetilde{H}(N^2)$

2. Trace out all trapezoids intersecting S as we did before by an auxiliary structure:

- Each leaf of $\widetilde{H}(N^i)$ stores its adjacent trapezoids in $H(N^i)$

3. Build new edges between trapezoids in $H(N^i) \setminus H(N^{i+1})$ between trapezoids in $H(N^{i+1}) \setminus H(N^i)$

- *Split*: If a trapezoid f is split into, $g_1, \ldots, g_j$, $j \leq 4$, for $1 \leq l \leq j$, there is an arc from f to g_l.

- *Merge*: If g_1 and g_2 are merged into g, for each parent f of g_1 and g_2, there is an arc from f to g

Lemma 2.2

Locating a point p in a trapezoid δ in $H(N^i)$ takes $O(\log i)$ expected time using $\widetilde{H}(N^i)$

- Since each trapezoid has at most 4 childen, the time of location is proportional to the number of trapezoids in $\widetilde{H}(N^i)$ which contain p
- We charge an involved trapezoid to its creator. In other words, S^j is charged if and only if p is contained in an trapezoid in $H(N^j)$ adjacent to S^j.
- Since a trapezoid is adjacent to at most 4 segments and $S_1, S_2, \ldots, S_n$ is a random sequence of N, the probability in which S^j will be charged is at most $4/j$.
- Expected time of locating p in a trapezoid δ in $H(N^i)$ is at most $1 + \sum_{j=1}^{i} 4/j = O(\log i)$

Lemma 2.3

Inserting S^{i+1} into $\widetilde{H}(N^i)$ takes $O(\log i + k(i+1)/n^2)$ expected time

- Step 1 takes $O(\log i)$ expected time
- Step 2 and Step 3 take time proportional to the number of intersection between $H(N^i)$ and S^{i+1} (as we do with conflict lists)
- The expected number of intersections between $H(N^i)$ and S^{i+1} is $O(k(i+1)/n^2)$
  - The expected number of intersection between N^{i+1} is $O(k(i+1)^2/n^2)$.

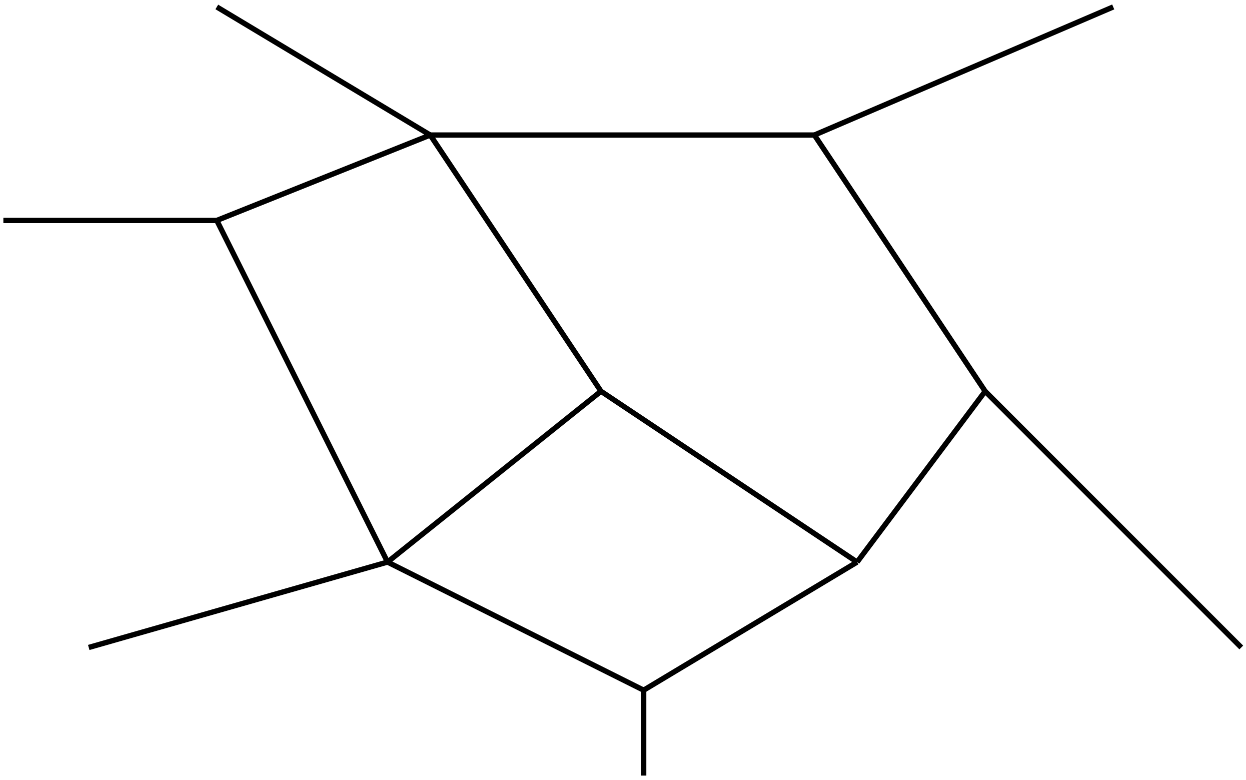
Theorem 2.2

Vertical trapezoidal composition formed by n segment in the plane can be computed in $O(k + n \log n)$ expected time by an on-line algorithm

- $\sum_{i=1}^{n} O(\log i + ki/n^2) = O(n \log n + k)$

Point Location Query:

Given a planar subdivision, process it such that for any query point q, the region to which q belong in the planar subidivision can be answered efficiently.



Solution: Let N be the edges of the planar subvisions.

1. Use history graph to compute the vertical trapezoidal decomposition $H(N)$ of N. Thus we have $\tilde{H}(N)$.
2. For a query point q, use $\tilde{H}(N)$ to answer to which in $H(N)$ trapezoid q belongs, and answer the region contains the trapezoid.

Time Complexity:

- Preprocessing Time: expected $O(n \log n)$
- Query Time : expected $O(\log n)$

Difference between conflict lists and history graph

- Conflict graph:
  the number of conflict relations between all trapezoids Δ in $H(N^i)$ adjacent to S^i and $N \setminus N^i$.
- History graph:
  the number of conflict relactions between S^i and trapezoids Δ in $\widetilde{H}(N^{i-1})$
- If S^i conflicts a trapezoid Δ created by S^j in $H(N^j)$, $j < i$, Δ and S^i form a conflict relaction in the conflict lists between $H(N^j)$ and $N \setminus N^j$
- The two total numbers are the same
- (S^i, Δ) is a conflict relation
  - Conflict Lists: charged when Δ is created
  - History Graph: charged when S^i is inserted.
- Conflict lists charge first, and history graph charges later.
- What not use history graph?