A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification
2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

MCS 481 Lecture 19
Computational Geometry
Jan Verschelde, 26 February 2025

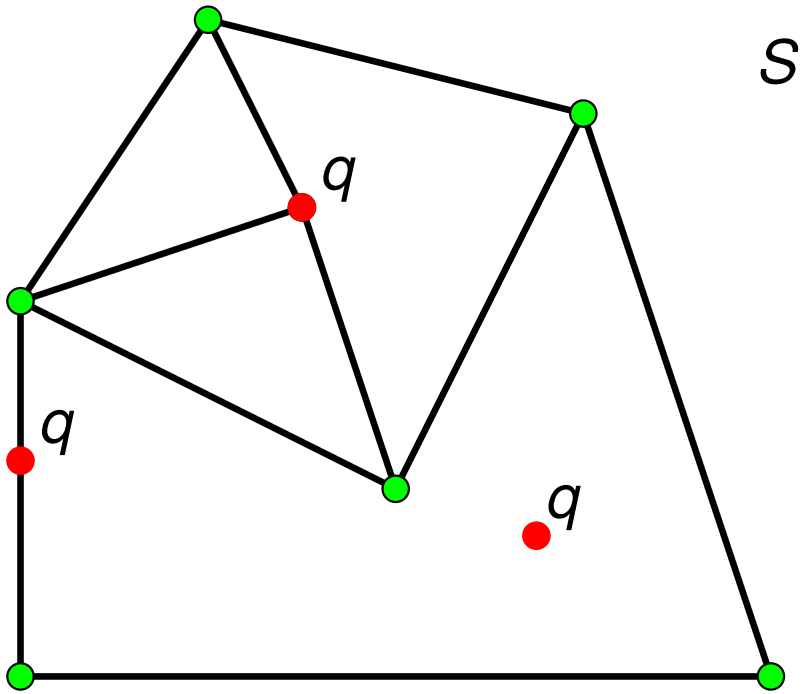
A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification
2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

the planar point location problem

Given a map, do you know where you are?

Input: a planar subdivision S, n = #edges of S,
a query point $q = (q_x, q_y)$.
Output: face of S that contains q, or eventually
the edge which contains q, or the vertex matching q.

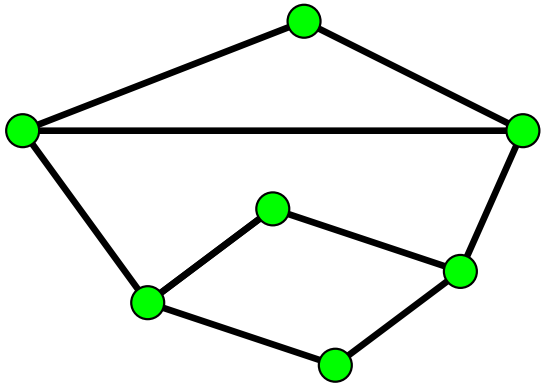


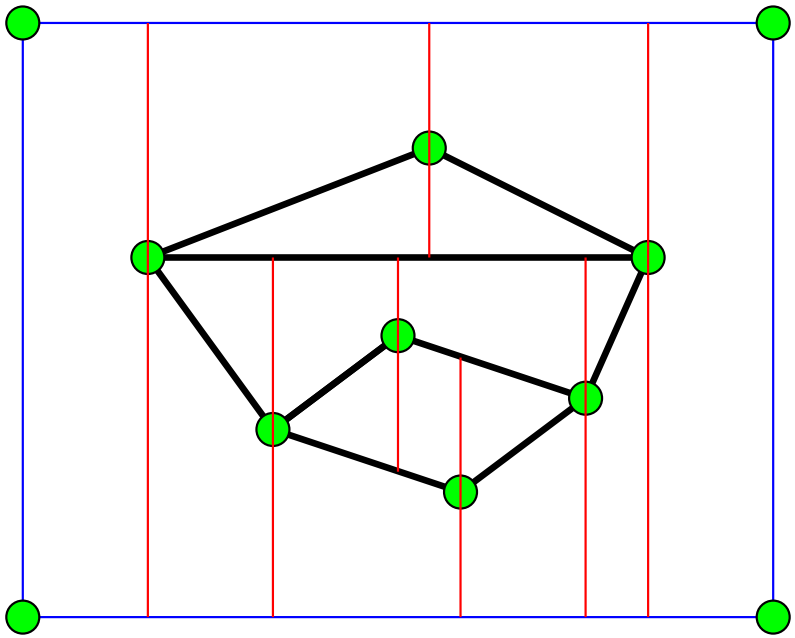
$S, n = 10$

3 cases:

- q in face
- q on edge
- q is vertex

a trapezoidal decomposition of a subdivision





Constructing a trapezoidal decomposition $T(S)$ of a subdivision S:

for every edge e in S do
    for every end point p of e do
        add a vertical extension from p to the next edge below p
        add a vertical extension from p to the next edge above p

A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification
2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

Trapezoidal Maps in CGAL

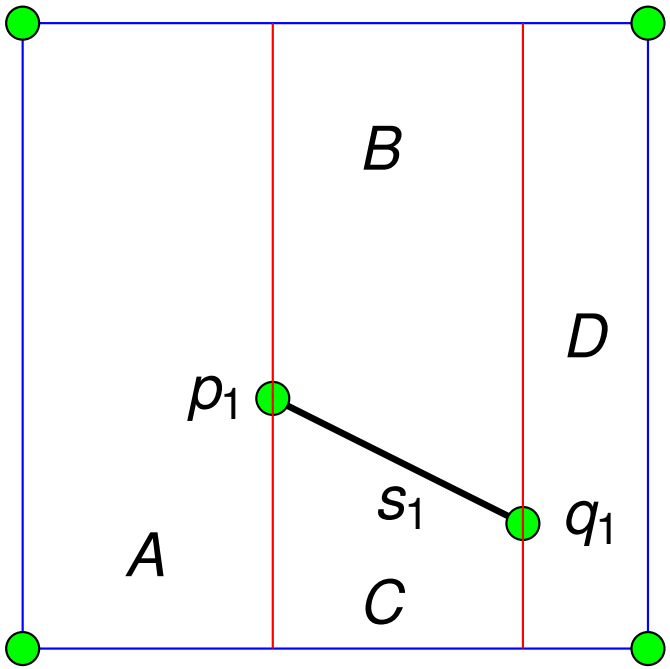
- Look in the tutorial on 2D Arrangements,
  in particular in the section Point-Location Queries.
- As the point-location strategy, select
  `Arr_trapezoid_ric_point_location<Arrangement>`
  which implements a variant of Ketan Mulmuley's algorithm.
- Ketan Mulmuley: **A Fast Planar Partition Algorithm, I**
  Journal of Symbolic Computation Vol. 10, pages 253-280, 1990.
- An example: `curve_history.cpp`.

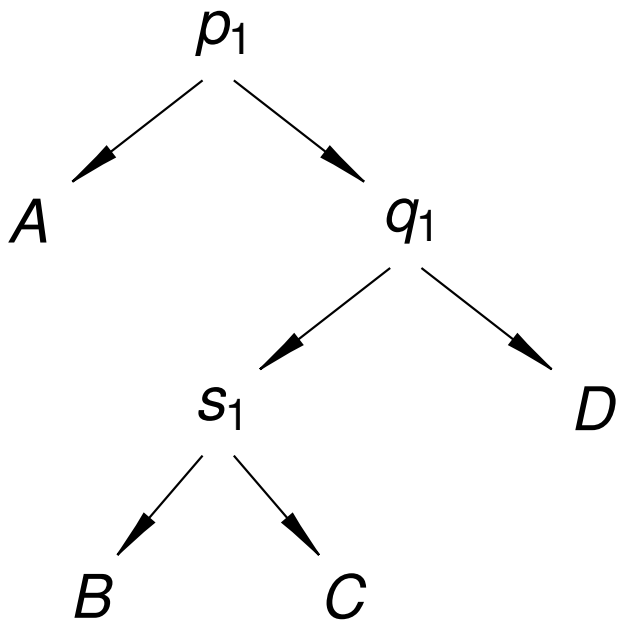
A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification
2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

searching a trapezoidal map

We use a directed acyclic graph to search a trapezoidal map.



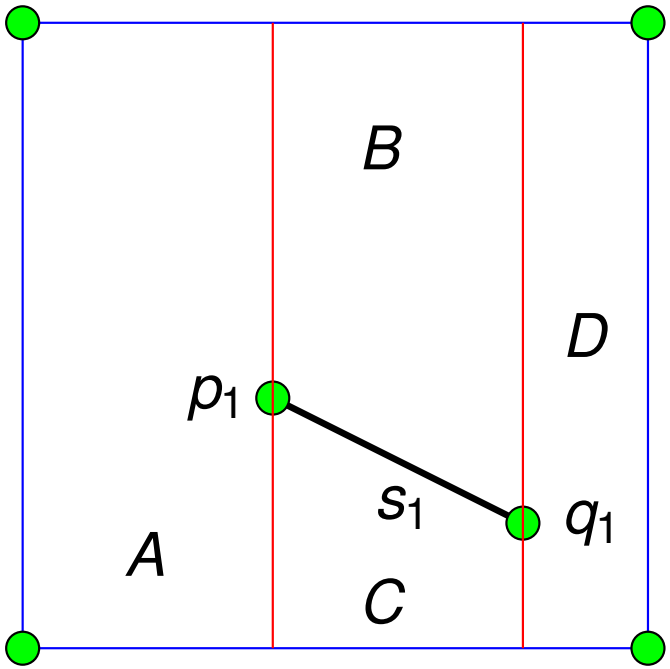


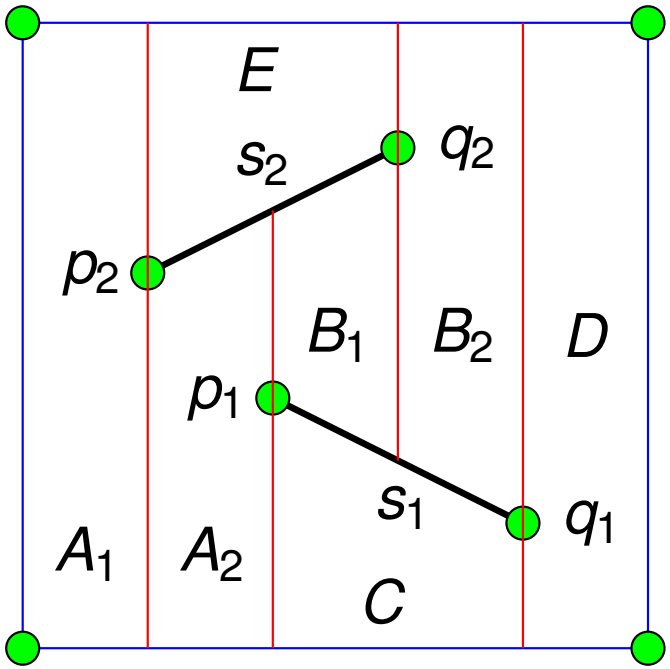
There are two types of nodes:

1. x-node: left or right of an end point of a line segment?
2. y-node: above or below a line segment?

Exercise 1: Write pseudo code for the above or below test.

update the trapezoidal map with a new line segment





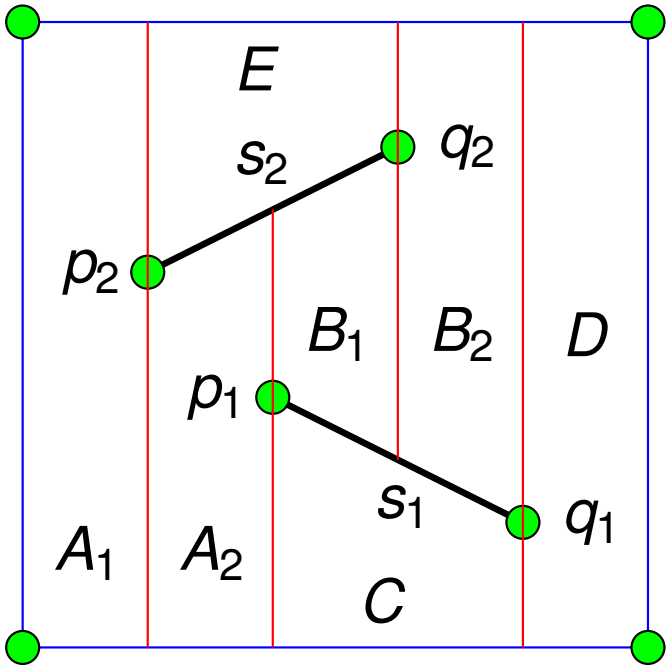
The new s_2 intersects the trapezoids A and B.

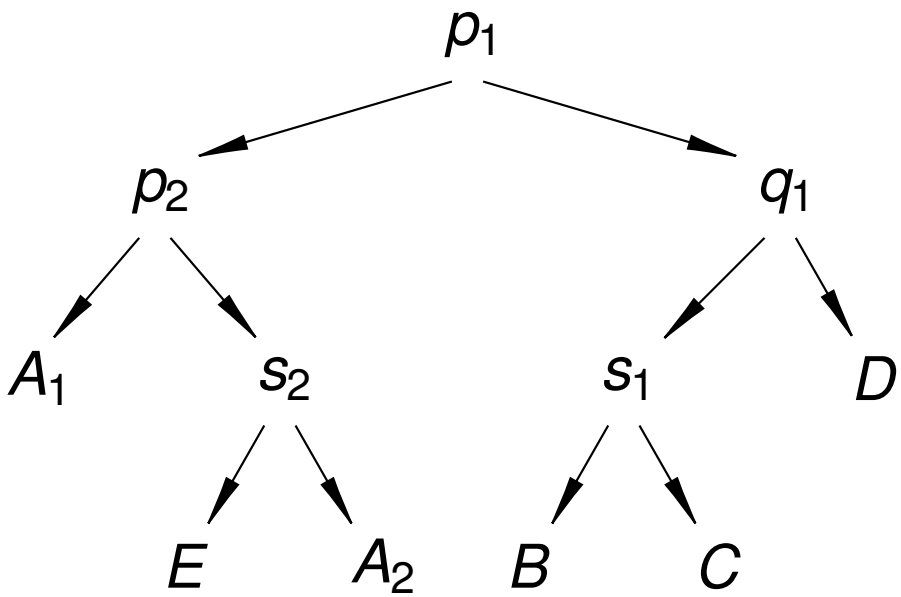
1. A is replaced by A_1, A_2, and E.
2. B is replaced by B_1, B_2, and E.

Observe the shortening of the extension through p_1.

update the graph with a new line segment

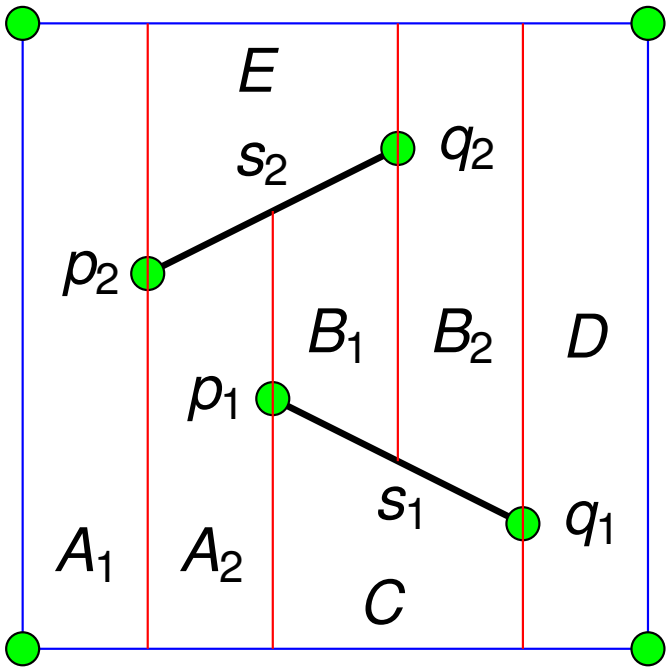
In the directed acyclic graph,
the leaf A, at the left of p_1, is replaced first:

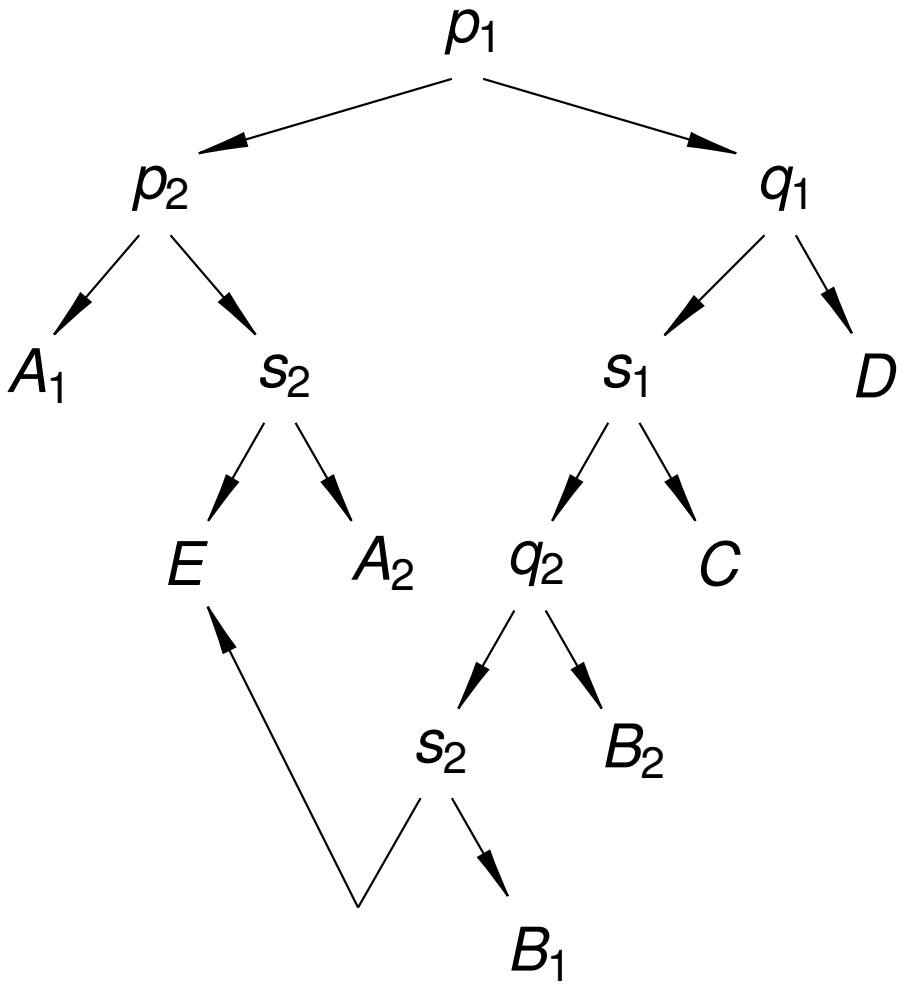




In the next step, the leaf B is replaced by B_1, B_2, and E.

after adding a line segment





In the directed acyclic graph, every trapezoid is stored only once.

The construction replaces leaves intersected by the new segment.

A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification
2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

input-output specification

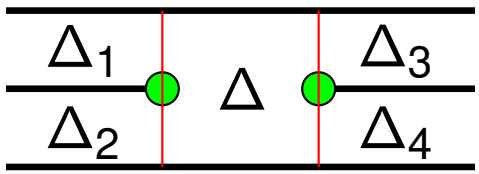
Algorithm TRAPEZOIDALMAP(S)

Input: S is a subdivision with n line segments.
Output: $T(S)$ is a list of trapezoids,
$\mathcal{D}$ a directed acyclic graph to search $T(S)$.

Assumptions on the input:

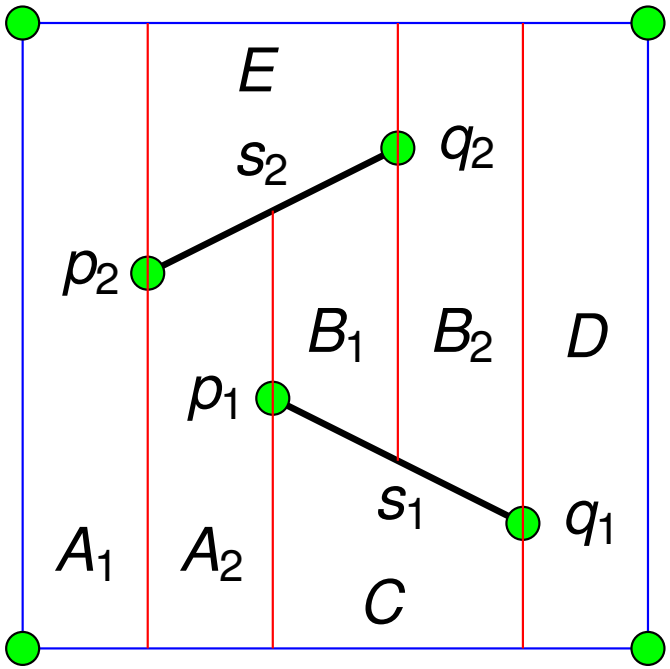
1. S is bounded by a rectangular box R,
2. S is in general position: no vertical segments, and no two points with the same x-coordinate.

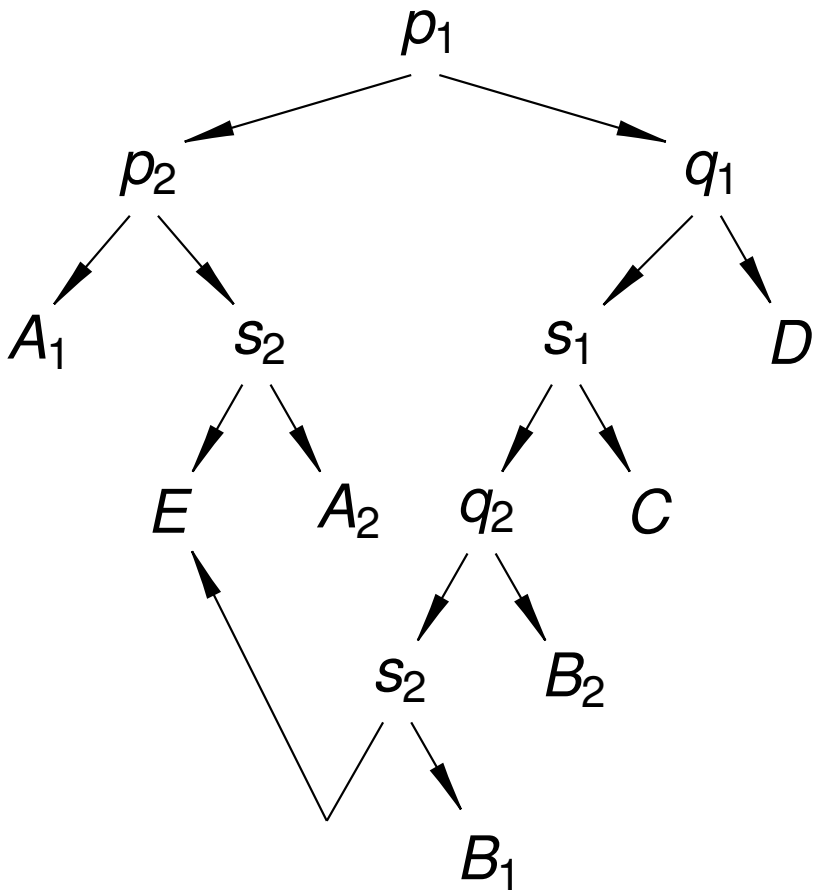
Every trapezoid $\Delta \in T(S)$ has at most 4 neighbors:



Every Δ is determined by $\mathrm{leftp}(\Delta)$, $\mathrm{rightp}(\Delta)$, $\mathrm{top}(\Delta)$, and $\mathrm{bottom}(\Delta)$.

the depth of the directed acyclic graph





Exercise 2: For n line segments, the depth of the directed acyclic graph $\mathcal{D}$ can be $O(n)$. Describe an (unfortunate) order of n line segments where the depth is $O(n)$.

A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification

2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

a randomized incremental algorithm

Algorithm TRAPEZOIDALMAP(S)

Input: S is a subdivision with n line segments.
Output: $T(S)$ is a list of trapezoids,
$\mathcal{D}$ a directed acyclic graph to search $T(S)$.

In an incremental algorithm,
the line segments are added one after the other.

For every new line segment s:

1. use $\mathcal{D}$ to locate trapezoids intersected by s,
2. update $T(S)$ with new trapezoids.

In a randomized incremental algorithm,
the order of adding the line segments is at random.

main steps in the algorithm

Algorithm TRAPEZOIDALMAP(S)

Input: S is a subdivision with n line segments.
Output: $T(S)$ is a list of trapezoids,
$\mathcal{D}$ a directed acyclic graph to search $T(S)$.

1. $R = \text{BOUNDINGBOX}(S)$
2. $T(S_0) = R$; $\mathcal{D} = \text{VERTICES}(R)$
3. $s_1, s_2, \ldots, s_n = \text{PERMUTE}(S)$
4. for $i = 1, 2, \ldots n$ do
5. $\quad \Delta_0, \Delta_1, \ldots, \Delta_k = \text{FOLLOWSEGMENT}(T(S_{i-1}), \mathcal{D}, s_i)$
6. $\quad \Delta'_0, \Delta'_1, \ldots, \Delta'_\ell = \text{UPDATET}(T(S_{i-1}), \Delta_0, \Delta_1, \ldots, \Delta_k, s_i)$
7. $\quad \text{UPDATED}(\mathcal{D}, \Delta_0, \Delta_1, \ldots, \Delta_k, \Delta'_0, \Delta'_1, \ldots, \Delta'_\ell, s_i)$

the loop invariant

1. $R = \text{BOUNDINGBOX}(S)$
2. $T(S_0) = R$; $\mathcal{D} = \text{VERTICES}(R)$
3. $s_1, s_2, \ldots, s_n = \text{PERMUTE}(S)$
4. for $i = 1, 2, \ldots n$ do
5. $\quad \Delta_0, \Delta_1, \ldots, \Delta_k = \text{FOLLOWSEGMENT}(T(S_{i-1}), \mathcal{D}, s_i)$
6. $\quad \Delta'_0, \Delta'_1, \ldots, \Delta'_\ell = \text{UPDATET}(T(S_{i-1}), \Delta_0, \Delta_1, \ldots, \Delta_k, s_i)$
7. $\quad \text{UPDATED}(\mathcal{D}, \Delta_0, \Delta_1, \ldots, \Delta_k, \Delta'_0, \Delta'_1, \ldots, \Delta'_\ell, s_i)$

The loop invariant:

1. $T(S_i)$ is a trapezoidal map for $s_1, s_2, \ldots, s_i$, and
2. $\mathcal{D}$ is a valid search structure for $T(S_i)$.

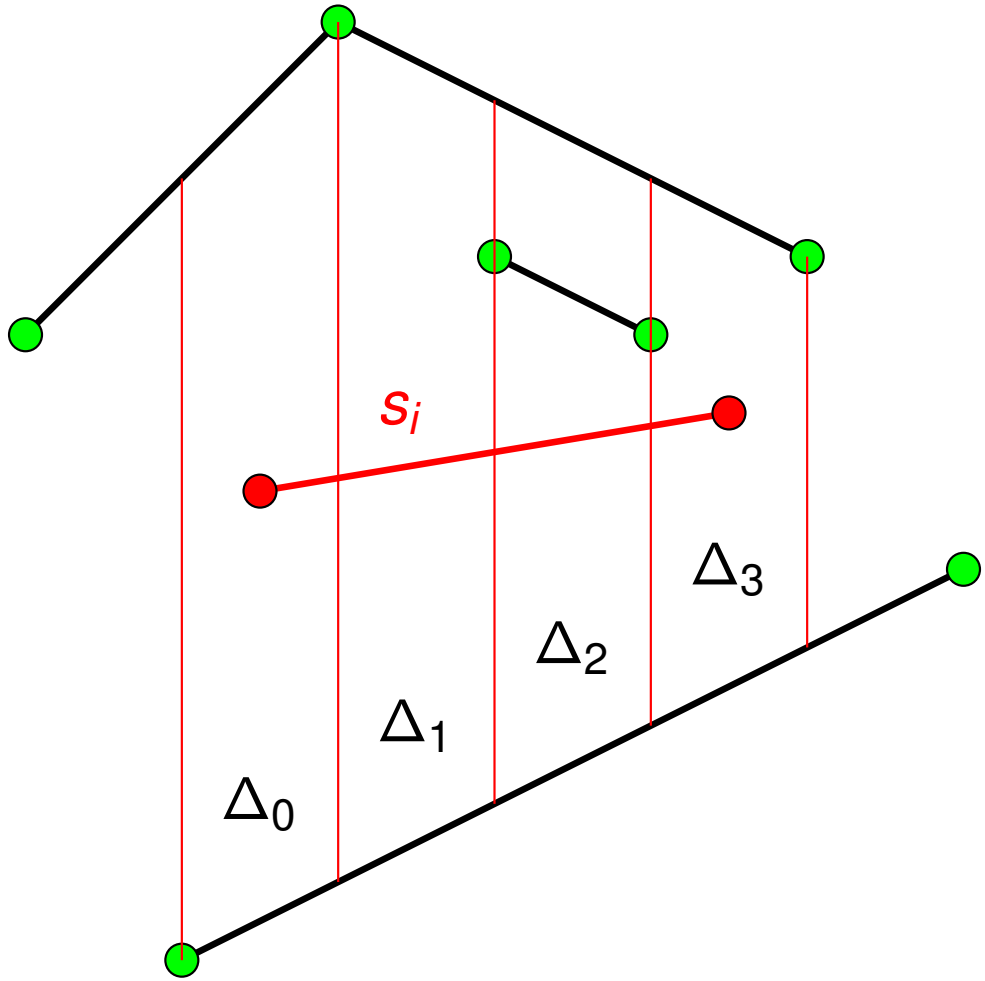
The correctness of the algorithm TRAPEZOIDALMAP follows from the correctness of FOLLOWSEGMENT, UPDATET, and UPDATED.

A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification

2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

locating intersected trapezoids



Exercise 3: Draw a directed acyclic graph *before* s_i was considered.

the algorithm to locate intersected segments

Algorithm FOLLOWSEGMENT($T, \mathcal{D}, s_i$)

Input: $T = T(S_{i-1})$, $\mathcal{D}$ is search DAG for T,
line segment s_i with end points p and q.
Output: $\{\Delta_0, \Delta_1, \ldots, \Delta_k\}$, $\Delta_j \cap s_i \neq \emptyset$, $j = 0, 1, \ldots, k$.

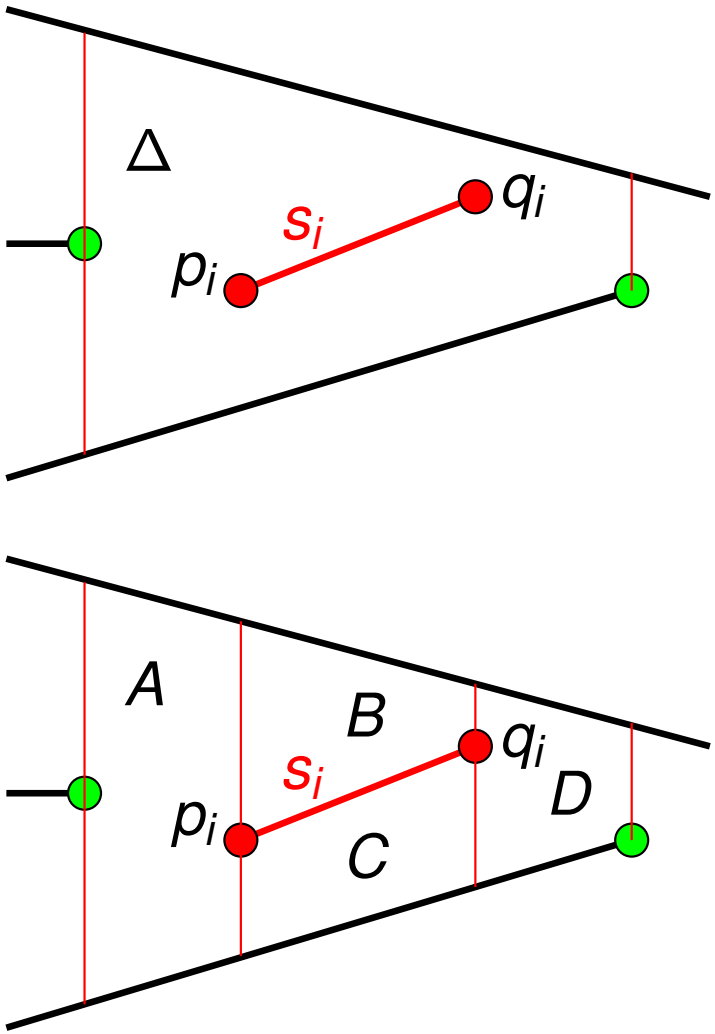
1. $\Delta_0 = \text{SEARCH}(\mathcal{D}, p)$; $j = 0$
2. while q at the right of $\text{rightp}(\Delta_j)$ do
3. if $\text{rightp}(\Delta_j)$ is above s_i
4. then $\Delta_{j+1} =$ lower right neighbor of Δ_j
5. else $\Delta_{j+1} =$ upper right neighbor of Δ_j
6. $j = j + 1$
7. return $\Delta_0, \Delta_1, \ldots, \Delta_j$

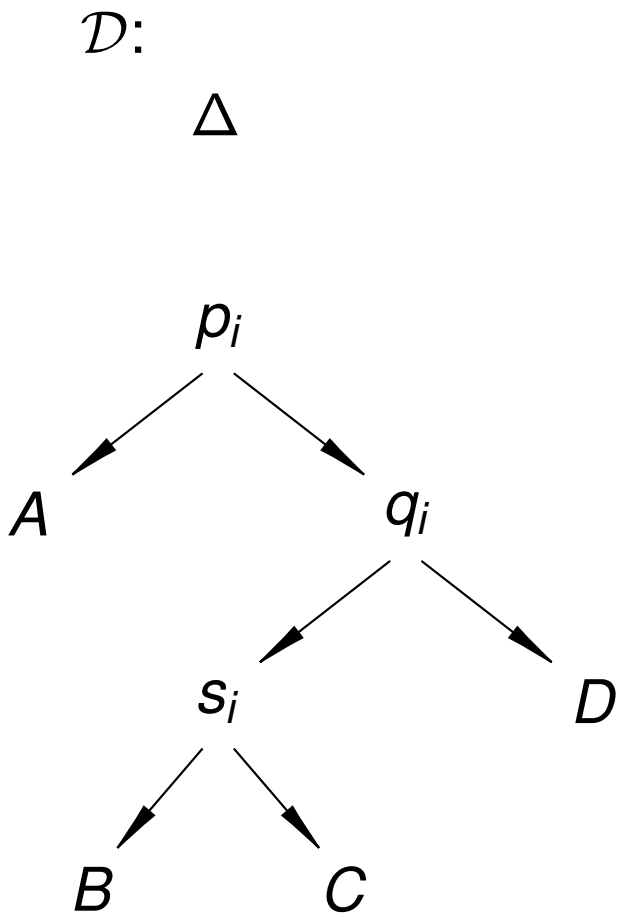
Observe that the adjacency information stored at each Δ is needed.
Exercise 4: Run the algorithm on the example on the previous slide.

A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification
2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

the segment s_i lies entirely inside one trapezoid



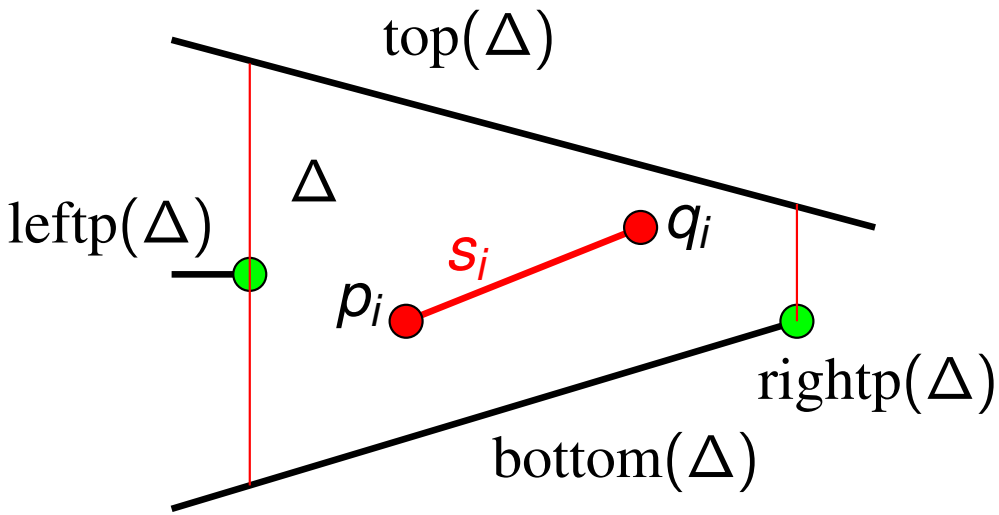


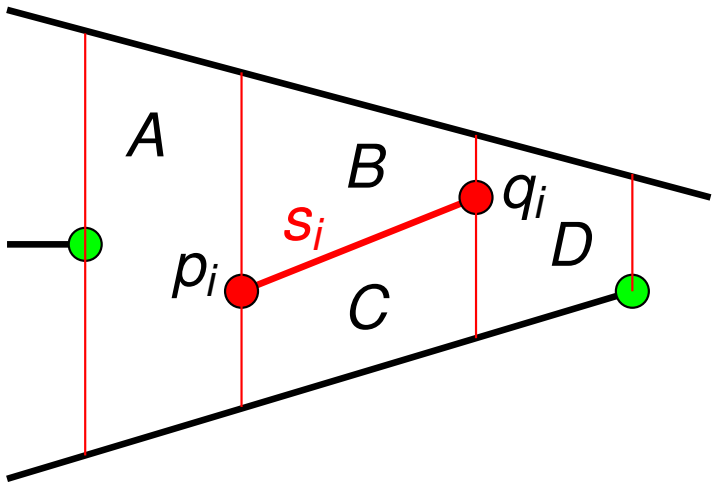
The trapezoid Δ is a leaf in $\mathcal{D}$ and is replaced by A, B, C, and D.

assigning top, bottom, leftp, and rightp

Every trapezoid Δ is determined by

1. top and bottom edges: $\text{top}(\Delta)$ and $\text{bottom}(\Delta)$,
2. left and right vertices: $\text{leftp}(\Delta)$ and $\text{rightp}(\Delta)$.

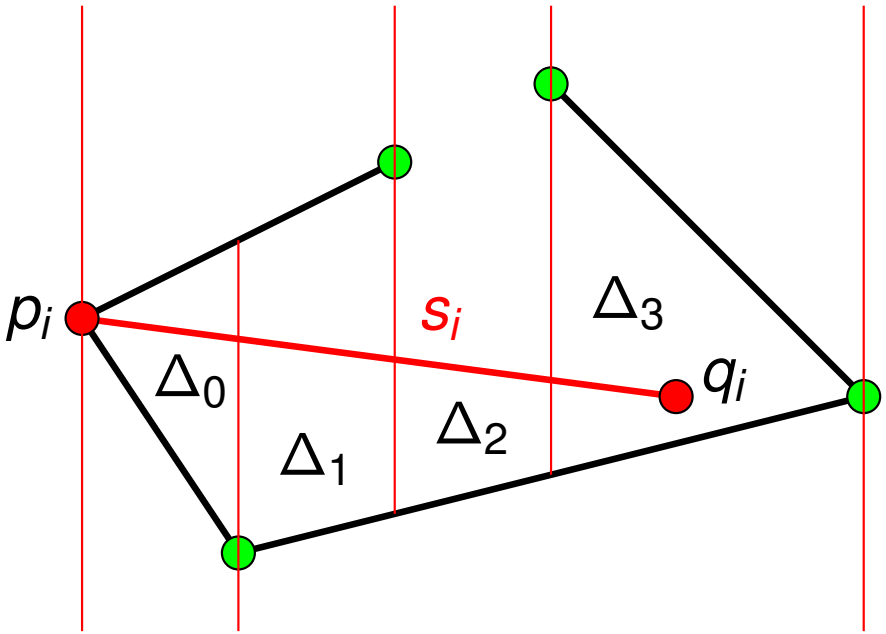


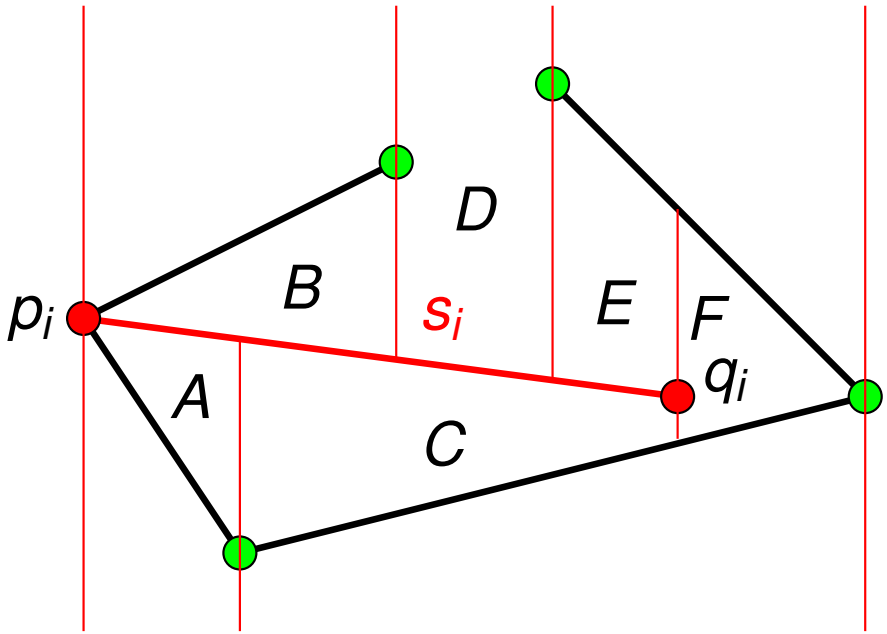


Exercise 5: For each new trapezoid A, B, C, and D, define $\text{top}()$, $\text{bottom}()$, $\text{leftp}()$, $\text{rightp}()$, in function of Δ, p_i, q_i, and s_i. Write your answer as a function in pseudo code.

the segment s_i intersects several trapezoids

s_i intersects Δ_0, Δ_1, Δ_2, and Δ_3.





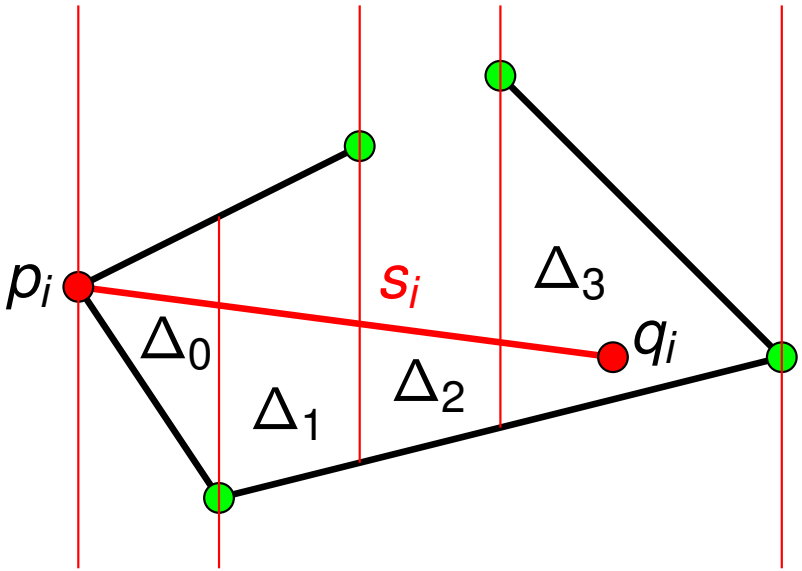
1. Define vertical extensions through p_i and q_i.
2. Shorten existing extensions so they end at s_i:
   new trapezoids have their top or bottom edge set to s_i.
3. Replace Δ_0, Δ_1, Δ_2, and Δ_3 by A, B, C, D, E, and F.

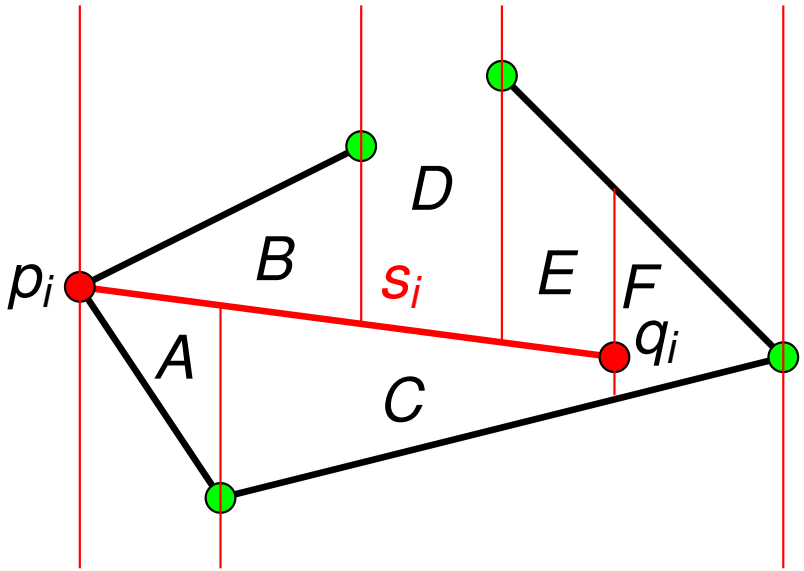
pseudo code for UPDATET

Exercise 6: Write pseudo code for UPDATET.

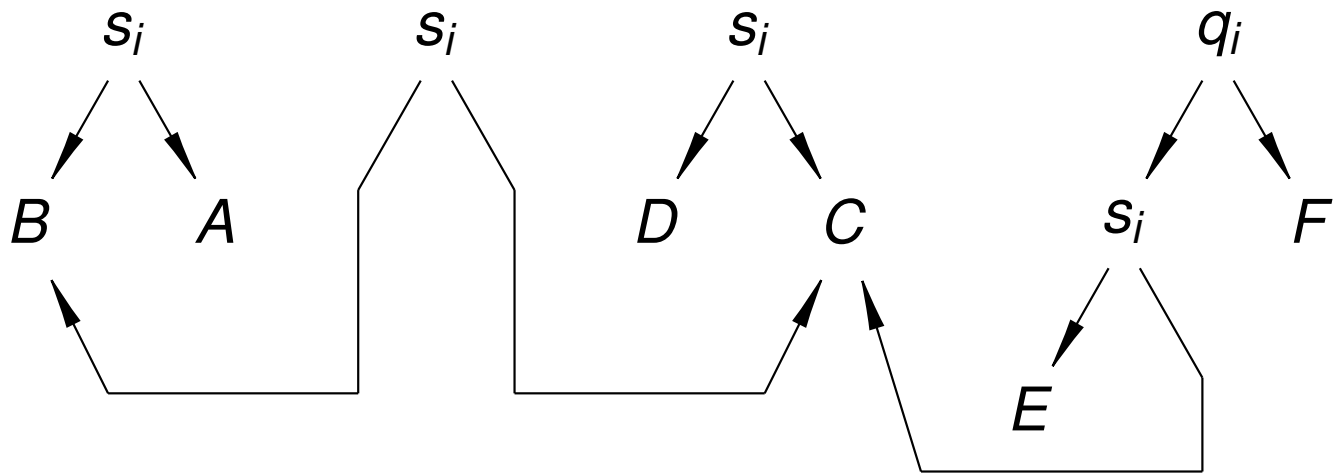
1. Define in detail the input-output specification.
2. Formalize the examples on the previous slides into code.

updating the search structure





Leaves Δ_0, Δ_1, Δ_2, and Δ_3 are replaced:



pseudo code for UPDATED

Exercise 7: Write pseudo code for UPDATED.

1. Define in detail the input-output specification.
2. Formalize the examples on the previous slides into code.

A Randomized Incremental Algorithm

1. Trapezoidal Maps
   - the planar point location problem
   - point location queries in CGAL
   - a directed acyclic graph
   - input-output specification

2. A Randomized Incremental Algorithm
   - adding line segments in a random order
   - locating intersected trapezoids
   - algorithms to update the data structures
   - statement of expected cost

statement of expected cost

Given is a subdivision S with n edges.

In the randomized algorithm, there are $n!$ ways to add the edges.

The expected cost is the average cost over all $n!$ running times.

Theorem (expected cost of a randomized incremental algorithm)

Given is a subdivision S with n edges in general position.

1. *A trapezoidal map $T(S)$ and a search structure $\mathcal{D}$ can be computed in $O(n\log(n))$ expected time.*
2. *The expected size of the search structure $\mathcal{D}$ is $O(n)$.*
3. *For any query point q, the expected query time is $O(\log(n))$.*

Next lecture will be devoted to a probabilistic analysis
of algorithm TRAPEZOIDALMAP, and
to remove the general position assumption.

summary and exercises

With a trapezoidal map and a directed acyclic graph we solve the point location problem.

We covered half of section 6.2 in the textbook.

Consider the following activities, listed below.

1. Write the solutions to exercises 1 through 7.
2. Consult the CGAL documentation and example code on trapezoidal maps.
3. Consider the exercises 5, 7, 8 in the textbook.