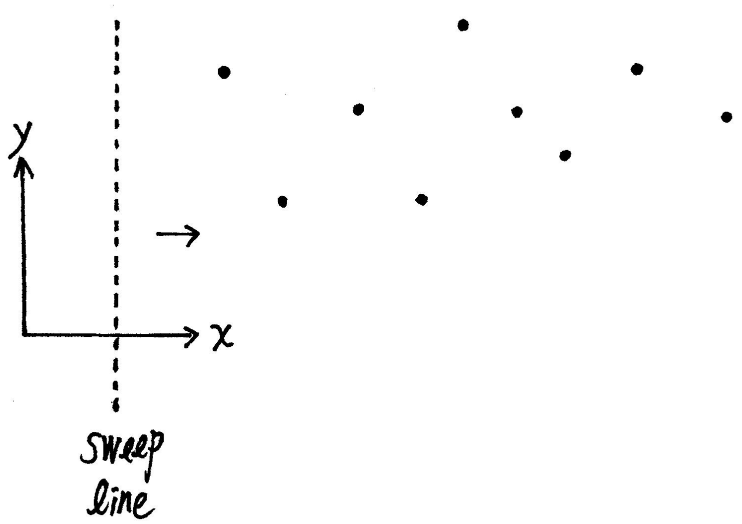
• Plane Sweep

Given a problem instance with a set of geometric objects, the plane sweep scans these objects from the left to the right.



These objects are collected in a data structure, called the *x-structure*.

When the sweep line stays at some object, the current status (i.e., all information needed to solve the problem) of the sweep is kept in a data structure, called the *y-structure*.

The plane sweep paradigm is sketched as follows.

1. **Initialize the *x*-structure and the *y*-structure.**

2. **While the *x*-structure is not empty, do**

    2.1 **Sweep to the object with minimal *x*-coordinate in the *x*-structure.**

    2.2 **Update the *y*-structure.**

Ex. The Line Segment Intersection Problem.

Given n line segments L_1, L_2, …, L_n in the plane, it is required to report their pairwise intersections.

Two assumptions :

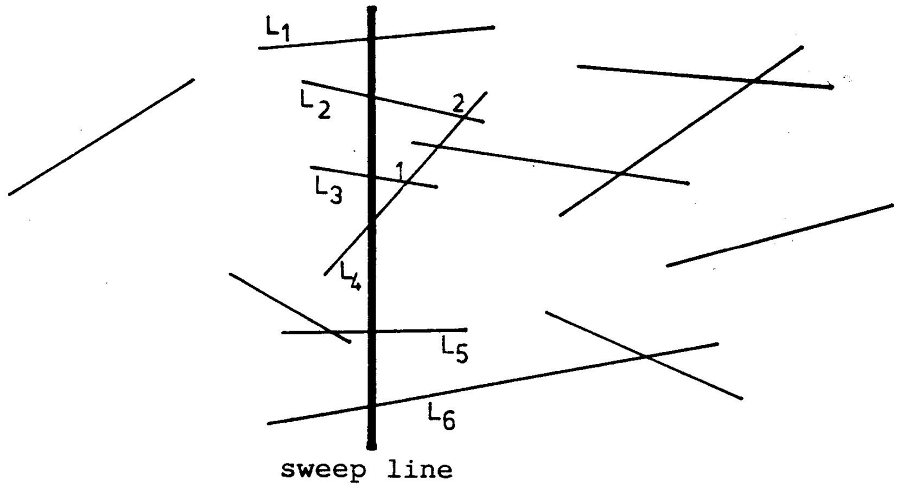
- **(1) no vertical line segment;**
- **(2) no common x-coordinate for all endpoints and intersection points.**

Initially, the x-structure contains the endpoints of all n line segments, maintained in an increasing sequence of their x-coordinates.

In run time, intersection points will be inserted into the x-structure.

The y-structure contains those line segments, ordered according to their y-coordinates that the sweep line intersects with.

The x-structure is implemented as a heap, and the y-structure is implemented as a balanced tree.

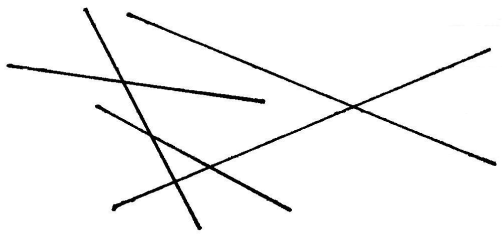


the y-structure : $L_1, L_2, L_3, L_4, L_5, L_6$

If we can collect all intersection points into the *x*-structure, then we can successfully report them after the sweep.

But, how to find all intersection points during the sweep ?

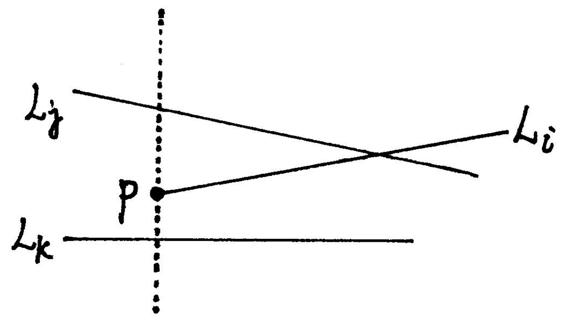
Observation : **If L_1 and L_2 intersect, then there is a point (endpoint or intersection point) on the left of their intersection, after which L_1 and L_2 are adjacent in the y-structure.**



Whenever the sweep line moves to the next point, denoted by p (i.e., p has the minimal x-coordinate in the current x-structure), both the x-structure and the y-structure are modified as follows.

1. **Delete p from the x-structure.**

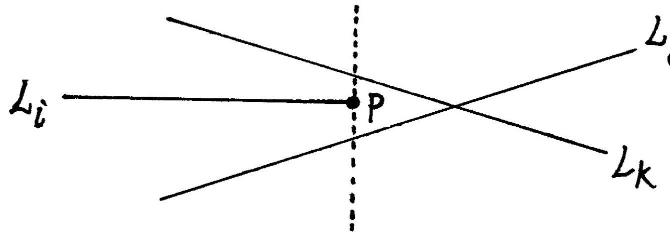
2. **If p is a left endpoint of some L_i, then**



(2-1) Insert L_i into the y-structure.

(2-2) Suppose that L_j and L_k are two neighbors of L_i in the y-structure. Insert the intersections of (L_i, L_j) and (L_i, L_k) into the x-structure, if they exist.

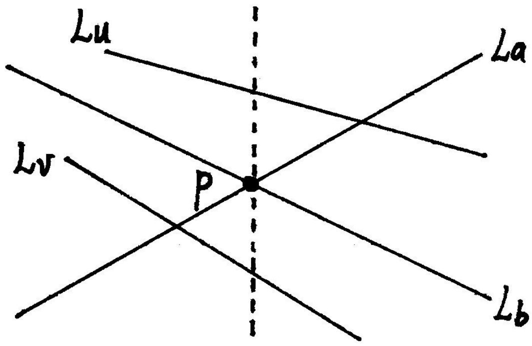
3. If p is a right endpoint of L_i, then



(3-1) Suppose that L_j and L_k are two neighbors of L_i in the y-structure. Insert the intersection of (L_j, L_k) into the x-structure, if it is on the right of the sweep line.

(3-2) Delete L_i from the y-structure.

4. **If p is the intersection of L_a and L_b, then**



(4-1) Interchange L_a and L_b in the y-structure.

(4-2) Suppose that L_u and L_v are two neighbors of L_a and L_b in the y-structure. Insert the intersections of (L_u, L_a) and (L_b, L_v) into the x-structure, if they exist and on the right of the sweep line.

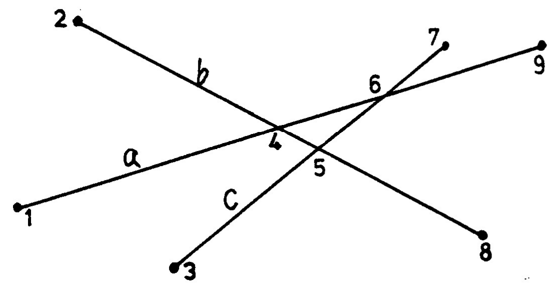
The algorithm needs to process $2n + s$ points, where s is the number of intersection points.

Since $O(\log(2n + s)) = O(\log n)$ time is required for each point, the total execution time for the algorithm is $O((n + s)\log n)$.

The space requirement is $O(n + s)$, and it can be further reduced to $O(n)$ (refer to: Bentley and Ottmann, "Algorithms for reporting and counting geometric intersections," *IEEE Trans. on Comput.*, vol. C-28, pp. 643-647, 1979.

An illustrative example.

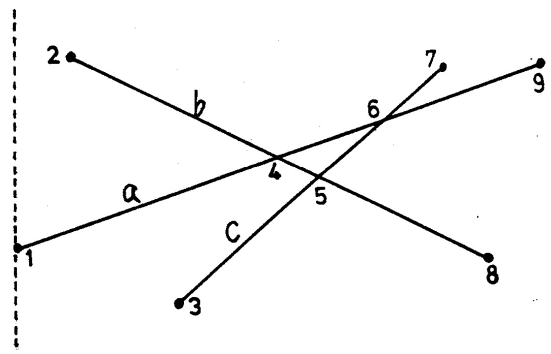
1. Initialization.



X-structure: 1, 2, 3, 7, 8, 9

Y-structure: ϕ

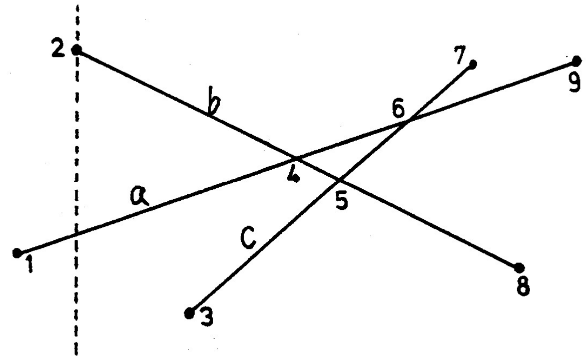
2. The sweep line stays at 1.



X-structure: 2, 3, 7, 8, 9

Y-structure: a

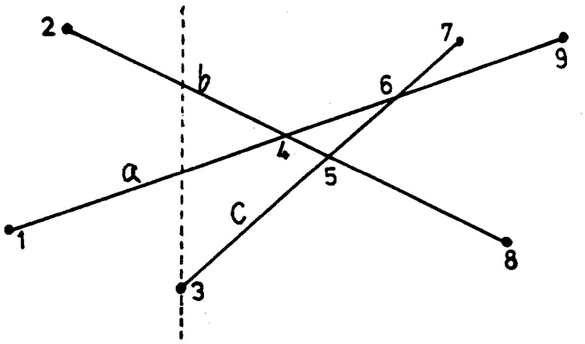
3. The sweep line stays at 2.



X-structure: 3, 4, 7, 8, 9

Y-structure: a, b

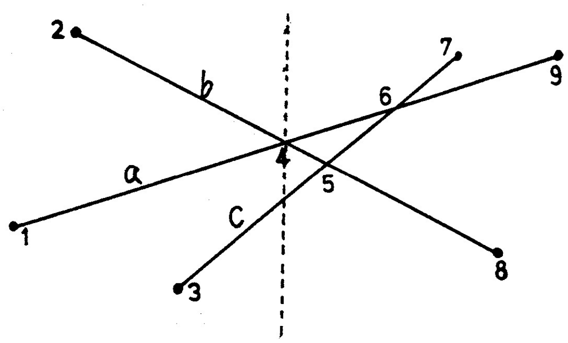
4. The sweep line stays at 3.



X-structure: 4, 6, 7, 8, 9

Y-structure: c, a, b

5. The sweep line stays at 4.

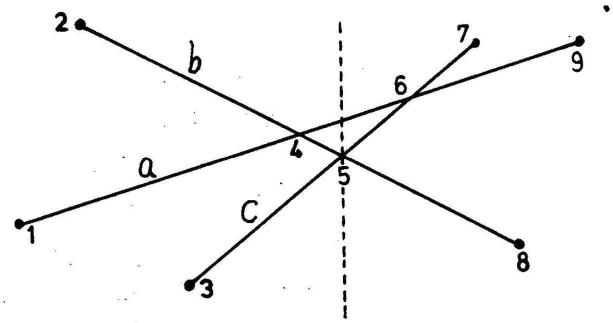


X-structure: 5, 6, 7, 8, 9

Y-structure: c, b, a

output 4

6. The sweep line stays at 5.

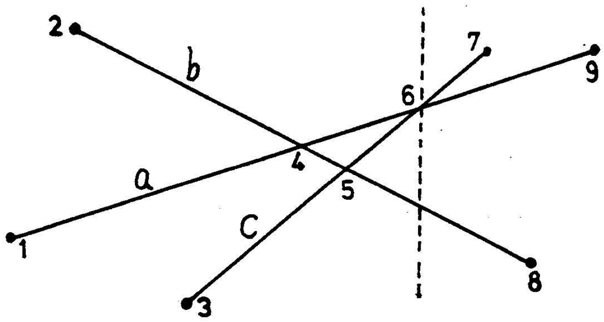


X-structure: 6. 7. 8. 9

Y-structure: b. c. a

output 5

7. The sweep line stays at 6.



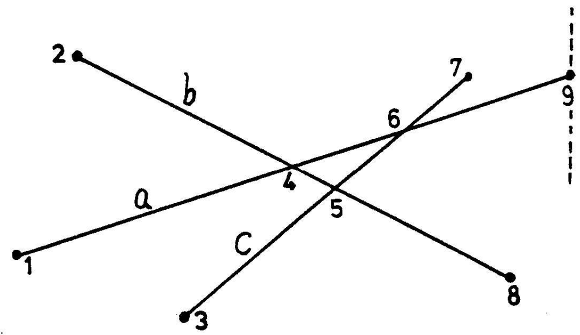
x-structure: 7, 8, 9

y-structure: b, a, c

output 6

⋮

10. The sweep line stays at 9.



x-structure: ϕ

y-structure: a

The best result thus far for the line segment intersection problem is $O(n*\log n + s)$ time and $O(n+s)$ space, which appears in "An optimal algorithm for intersecting line segments in the plane" (by Chazelle and Edelsbrunner, *J. ACM*, vol. 39, no. 1 pp. 1-54, 1992).