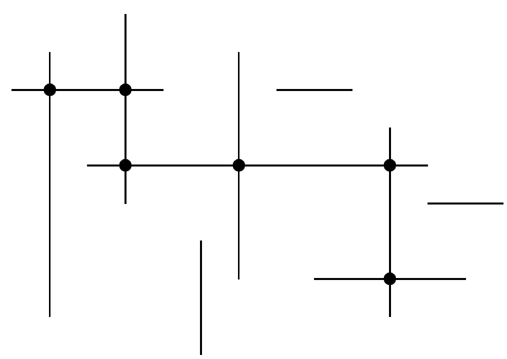
Computing intersections in a set of horizontal and vertical line segments

Michiel Smid*

October 14, 2003

We are given a set of horizontal and vertical line segments in the plane and want to compute all intersection points. The total number of line segments will be denoted by n.



As always, there is a very simple algorithm to solve this problem: For each pair of segments, test whether they intersect. (By the way, how do you do one such test?) The running time is of course $O(n^2)$. Moreover, this is optimal, because the number of intersections can be quadratic in n. Our goal is an algorithm whose running time does not only depend on n, but also on the number k of intersections. The algorithm should be "fast" when k is "small". Such an algorithm is called *output-sensitive*.

We will solve the problem using *plane sweep*. The idea is as follows: We move (or sweep) a vertical line SL (the *sweep line*) from left to right over the plane $\mathbb{R}^2$. During this sweep, the following invariant will be maintained:

*School of Computer Science, Carleton University, Ottawa, Ontario, Canada K1S 5B6. E-mail: michiel@scs.carleton.ca.

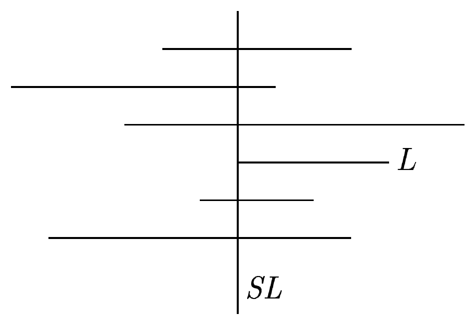
Invariant:

- All intersection points that are to the left of SL have been found already.
- All horizontal segments that intersect SL are stored in a balanced binary search tree, sorted by their y-coordinates.

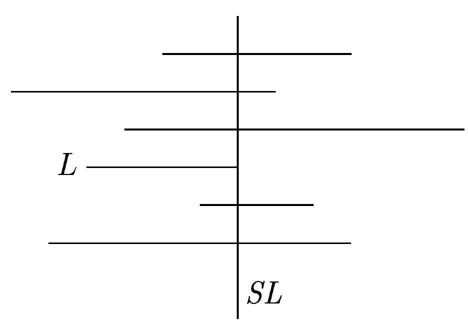
After the sweep line SL has visited all line segments (i.e., SL is to the right of all segments), then it follows from the invariant that all intersection points have been found.

What happens during the sweep? There are three cases to consider.

SL hits at the left endpoint of a horizontal line segment: Let L be this horizontal line segment. When the sweep line moves further to the right, L intersects SL. Therefore, in order to maintain the invariant, we insert L into the balanced binary search tree.

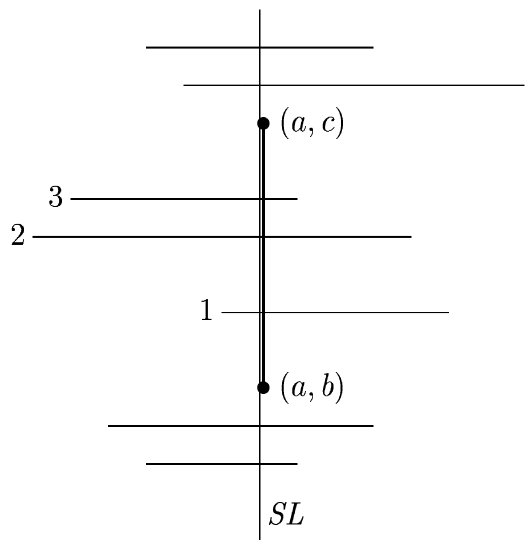


SL hits at the right endpoint of a horizontal line segment: Let L be this horizontal line segment. It follows from the invariant that L is stored in the balanced binary search tree. When the sweep line moves further to the right, L does not intersect SL any more. Therefore, in order to maintain the invariant, we delete L from the balanced binary search tree.



***SL* hits at a vertical line segment:** Let L be this vertical line segment. We search for all horizontal line segments that intersect L. Observe that these horizontal segments intersect the sweep line SL and, therefore, are stored in our balanced binary search tree.

Let (a, b) and (a, c), with $b < c$, be the endpoints of L. Then we search in the balanced binary search tree for all horizontal line segments whose y-coordinates are in the range $[b, c]$. (Observe that this is a one-dimensional orthogonal range query.) In the example below, the horizontal line segments 1, 2, and 3 are reported.



The complete algorithm is presented in Figure 1. It is not difficult to see that the running time is $O(n \log n + k)$, where n is the number of input line segments and k is the number of intersection points.

Algorithm *Intersections*(H, V)
(* H is a set of horizontal line segments,
  V is a set of vertical line segments *)
sort the x-coordinates of the left and right endpoints of the
segments of H and the x-coordinates of the segments in V, and
store the sorted sequence in an array $A[1 \ldots N]$;
$B :=$ empty balanced binary search tree;
for $i := 1$ **to** N
do let L be the segment that corresponds to $A[i]$;
  if $L \in H$ and $A[i] =$ left endpoint of L
  then insert (the y-coordinate of) L into B
  else if $L \in H$ and $A[i] =$ right endpoint of L
    then delete (the y-coordinate of) L from B
    else let (a, b) und (a, c) be the endpoints of L, where $b < c$;
      find all segments in B whose y-coordinates are in $[b, c]$
    endif
  endif
endfor

Figure 1: *The algorithm that computes all intersections in a set of horizontal and vertical line segments.*