Roads solution

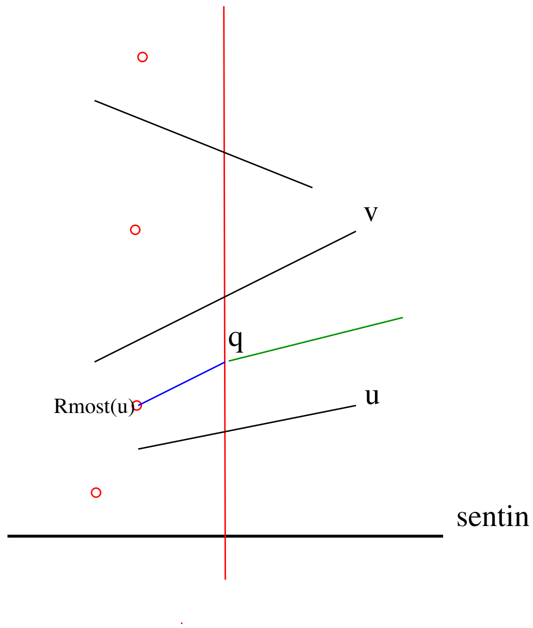
We assume that for segment $s = (p, q)$ the relation $p.x < q.x$ or $p.x = q.x$ and $p.y < q.y$ holds, therefore we can say that p is the left endpoint of the segment.
Consider the sequence of segment endpoints ordered by their x-coordinates.
We apply sweep-line method, the event points of the sweeping are the x-coordinates of the endpoints.
For a given sx coordinate let us denote by $Sl(sx)$ the set of segments intersecting the vertical line whose x coordinate is sx. Element of the set Sl are ordered according to the y-coordinate of the intersection points. As sweep progress from let to right, if the point is left endpoint than it is inserted into Sl, and deleted if it is right endpoint.
Our model solution is based on the following observations.
We add a sentinel, as figure shows. For each segment $u \in Sl$ we compute a segment endpoint $Rmost(u)$.

circuit invariant

1. Each segment endpoint which is on the right of the sweep-line already a node of the tree, the partial output.

2. Let $u \in Sl$ and $v \in Sl$ be segments such that v directly next to u according to the ordering. Then for every point q on the sweep-line located between the intersection points of u and v, if the $\overline{Rmost(u), q}$ intersects any old or newly added segment then the intersection point must be a point of a city (endpoint of an old segment).

Therefore if we insert a left endpoint q into Sl then segment $(Rmost(u), q)$ will be added to the solution.
Moreover, during both insertion and deletion we can update $Rmost(u)$ value in $O(log\, N)$ time if we represent the set Sl by STL ordered set.
The running time of the whole algorithm is $O(N \log N)$.



Subtask 1

Constraint: All segments are vertical.
This subtask can be solved by sorting the segments and connecting consecutive segments left and right endpoint. The sorting relation is easy to compute: $s_1 = (p_1, q_1) < s_2 = (p_2, q_2)$ iff $p_1.x < p_2.x$ or $p_1.x = p_2.x$ and $p_1.y < p_2.y$.

Subtask 2

Constraint: Each pair of segments are parallel.
This subtask also admits solution by sorting, but the computation of the sorting relation is not so easy. Namely, $s_1 = (p_1, q_1) < s_2 = (p_2, q_2)$ iff both p_1 and q_1 located on the left of the line determined by s_2 or s_1 and s_2 are colinear and q_1 is in between p_2 and q_2.

Subtask 3

Constraint: Each segment is either horizontal or vertical.
This subtask can be solved by simple implementation of the model solution algorithm, because it is easy to compute the sorting relation of the set Sl.

Subtask 4

Constraint: $N \leq 1000$
There is an $O(N^2)$running time algorithm that can solve this subtask. For example if we represent the sets $Sl(sx)$ by sorted array.