A Simple Algorithm for Computing the Zone of a Line in an Arrangement of Lines*

Haitao Wang†

Abstract

Let L be a set of n lines in the plane. The zone $Z(\ell)$ of a line ℓ in the arrangement $\mathcal{A}(L)$ of L is the set of faces of $\mathcal{A}(L)$ whose closure intersects ℓ. It is known that the combinatorial size of $Z(\ell)$ is $O(n)$. Given L and ℓ, computing $Z(\ell)$ is a fundamental problem. Linear-time algorithms exist for computing $Z(\ell)$ if $\mathcal{A}(L)$ has already been built, but building $\mathcal{A}(L)$ takes $O(n^2)$ time. On the other hand, $O(n \log n)$-time algorithms are also known for computing $Z(\ell)$ without relying on $\mathcal{A}(L)$, but these algorithms are relatively complicated. In this paper, we present a simple algorithm that can compute $Z(\ell)$ in $O(n \log n)$ time. More specifically, once the sorted list of the intersections between ℓ and the lines of L is known, the algorithm runs in $O(n)$ time. A big advantage of our algorithm, which mainly involves a Graham's scan style procedure, is its simplicity.

1 Introduction.

Given a set L of n lines in the plane, let $\mathcal{A}(L)$ denote the arrangement of the lines of L, i.e., the subdivision of the plane induced by L. For a line ℓ, the *zone* of ℓ in the arrangement $\mathcal{A}(L)$ is the set of faces of $\mathcal{A}(L)$ whose closure intersects ℓ (see Fig. 1); we use $Z(\ell)$ to denote the zone of ℓ. Given L and ℓ, we consider the problem of constructing $Z(\ell)$.

It has been proved that the combinatorial size of the zone $Z(\ell)$ is bounded by $O(n)$ [5–7, 10, 12, 13]. The problem of computing $Z(\ell)$ is a fundamental problem in computational geometry. If the arrangement $\mathcal{A}(L)$ has already been explicitly constructed, then $Z(\ell)$ can be computed in $O(n)$ time [6, 12]. Indeed, this leads to an incremental algorithm for constructing the arrangement $\mathcal{A}(L)$ in $O(n^2)$ time [6, 12]. Without having the arrangement $\mathcal{A}(L)$, computing $Z(\ell)$ can be done in $O(n \log n)$ time. For example, Alevizos, Boissonnat, and Preparata [4] proved that the size of any cell in an arrangement of a set of n rays in the plane is $O(n)$ and gave an $O(n \log n)$ time algorithm to construct any cell. Their algorithm can be used to compute $Z(\ell)$ in $O(n \log n)$ time. Indeed, we can cut each line of L into two rays at its intersection with ℓ. Then, for each side of ℓ, we compute the cell containing ℓ in the arrangement of the rays on that side of ℓ. The zone $Z(\ell)$ can be obtained from the two cells computed above.

In this paper, we present a new algorithm for computing the zone $Z(\ell)$ in $O(n \log n)$ time. More specifically, once the sorted list of all intersections between ℓ and the lines of L is known, the algorithm runs in $O(n)$ time. In contrast, even if the above sorted list is known, applying the algorithm of [4] to compute $Z(\ell)$ still takes $O(n \log n)$ time because the algorithm involves sweeping line procedures that are modifications of the classical algorithm for computing the intersections of line segments. A big advantage of our algorithm is that it is quite simple. Indeed, a main process of our algorithm is a Graham's scan style procedure, which is a textbook level algorithm. As computing $Z(\ell)$ is a fundamental problem and many algorithms use it as a subroutine (e.g., [1, 2, 9]), it is worth pursuing a simple algorithm.

Many other problems in arrangements of lines or other curves are also fundamental and have been extensively studied. We refer the reader to [3, 7, 14, 15] for some excellent books and surveys.

2 The algorithm.

Without loss of generality, we assume that ℓ is the x-axis. To simplify the discussion, we make a general position assumption that no line of L is horizontal and no two lines of L have an intersection on ℓ. We will discuss at the end of this section that degenerate cases can be easily handled, although standard techniques [8, 11] could be applied too.

*This research was supported in part by NSF under Grant CCF-2005323.

†Department of Computer Science, Utah State University, Logan, UT 84322, USA. Email: haitao.wang@usu.edu

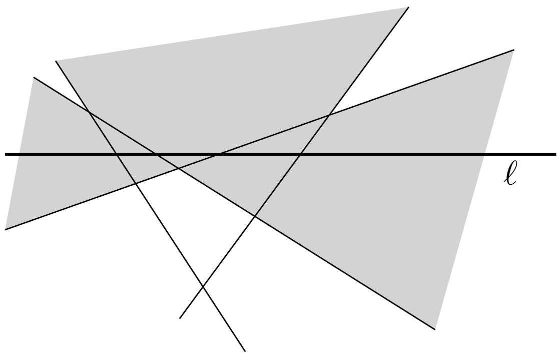

Figure 1: The shaded region is the zone $Z(\ell)$.

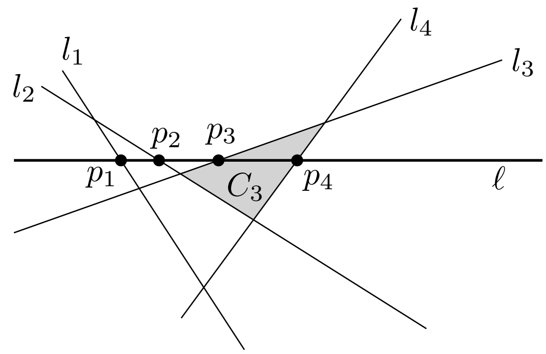

Figure 2: Illustrating the points p_i and a cell C_3.

A curve γ in the plane is *y-monotone* if any horizontal line either does not intersect γ or it intersects γ at a single point. We say that a y-monotone convex curve γ is *rightward* (resp. *leftward*) if γ always makes right turns from its lower endpoint to its upper endpoint.

Due to the general position assumption, every line of L intersects ℓ at a point. We start by computing the intersections between ℓ and all lines of L, and then sort them. This takes $O(n \log n)$ time. The rest of the algorithm runs in $O(n)$ time. Let $Z^+(\ell)$ denote the portion of the zone $Z(\ell)$ above ℓ and $Z^-(\ell)$ the portion of $Z(\ell)$ below ℓ. In the following, we describe an algorithm to compute $Z^+(\ell)$ in $O(n)$ time; $Z^-(\ell)$ can be computed in $O(n)$ time analogously.

Let $l_1, l_2, \ldots, l_n$ be the sorted list of the lines of L from left to right by their intersections with ℓ. Due to our general position assumption, this order is unique. For each $1 \leq i \leq n$, define p_i as the intersection of l_i and ℓ (see Fig. 2). The point p_i divides l_i into two half-lines, and we use l_i^+ to refer to the one above ℓ. Hence, p_i is the lower endpoint of l_i^+; for reference purpose, we also assume that l_i^+ has an upper endpoint at infinity and use p_i' to denote it. For convenience, let p_0 represent the left endpoint of ℓ at $-\infty$ and p_{n+1} the right endpoint of ℓ at ∞.

It is easy to see that for each $0 \leq i \leq n$, the segment $\overline{p_i p_{i+1}}$ is contained in a single cell of $\mathcal{A}(L)$, denoted by C_i, which is also a cell in $Z(\ell)$ (see Fig. 2). Denote by C_i^+ the portion of C_i above ℓ. Observe that $Z^+(\ell)$ is the disjoint union of cells C_i^+ for all $i = 0, 1, \ldots, n$. Hence, it suffices to compute the cells C_i^+ for all $i = 0, 1, \ldots, n$.

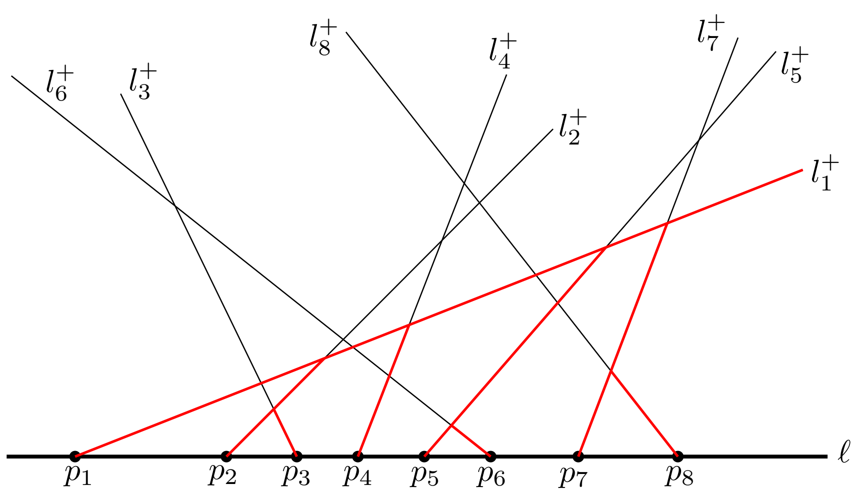

Figure 3: The forward forest F is colored red.

Forward and backward forests. With respect to the index order $1, 2, \ldots, n$, we define a *forward forest* F as follows (see Fig. 3). Let $F_1 = \ell$. For each $2 \leq i \leq n$, F_i is obtained from F_{i-1} by adding to F_{i-1} the segment of l_i^+ from p_i to its first intersection with F_{i-1} (if l_i does not intersect F_{i-1}, then the entire l_i^+ is included in F_i). Let $F = F_n$. It is not difficult to see that F has at most $2n-1$ edges because F_i has at most two more edges

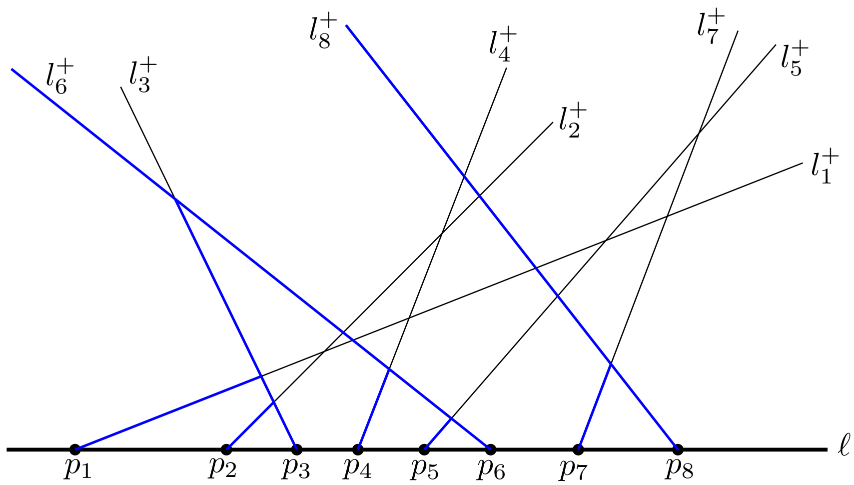

Figure 4: The backward forest F' is colored blue.

than F_{i-1}. We can view F as a forest in which the leaves are the points p_i for all $1 \leq i \leq n$ and the roots are upper endpoints of some half-lines l_i^+ (e.g., in Fig. 3, F consists of only one tree, whose root is the upper endpoint of l_1^+). That is why we call F a forest. Notice that if we move from a point p_i along F until the root of the tree containing p_i, we always turn rightward, i.e., the path from p_i to the root is a y-monotone rightward convex chain.

Similarly, we define a *backward forest* F' with respect to the inverse index order $n, n-1, \ldots, 1$ (see Fig. 4, where F' consists of two trees with roots at the upper endpoints of l_8^+ and l_6^+, respectively). The path from each leaf p_i to its root in F' is a y-monotone leftward convex chain

We remark that the forward/backward forest is very similar to the leftist/rightist skeleton in [4] as well as the upper/lower horizon tree in [8]. Note that it might also be possible to use the topologically sweeping method of [8] to construct $Z^+(\ell)$. However, the algorithm needs to dynamically update the upper/lower horizon trees after processing each event. In contrast, as will be seen later, our algorithm does not need to update the forests, which is not only simpler but also simple.

Forest decomposition and a Graham's scan style algorithm. We will use both forests F and F' to construct the cells C_i^+ for all $1 \leq i \leq n$. To this end, we describe a Graham's scan style algorithm to compute F (the other forest F' can be computed analogously). As will be seen, the algorithm also naturally decomposes F into n y-monotone rightward convex chains $\alpha_1, \alpha_2, \ldots, \alpha_n$, such that F is the edge-disjoint union of all these chains (i.e., each edge of F belongs to one and only one chain; see Fig. 5). For each i, the lower endpoint of α_i is p_i, and α_i is a sub-path of the path from p_i to the root of the tree containing p_i. Specifically, α_n is the path from p_n to its root in F. For each $1 \leq i \leq n-1$, α_i is the sub-path from p_i to its tree root until the first vertex in the union of the chains $\alpha_{i+1}, \alpha_{i+2}, \ldots, \alpha_n$.

Our algorithm will maintain a y-monotone rightward convex chain, denoted by α. Initially, we set $\alpha = l_1^+$. We process l_i^+ iteratively following the order $i = 2, 3, \ldots, n$. For $i = 2$, we first check whether l_2^+ intersects α. If yes (let q be the intersection of l_2^+ and α), then α_1 is defined as $\overline{p_1 q}$ and α is updated to the union of $\overline{p_2 q}$ and $\overline{q p_1'}$ (recall that p_1' denotes the upper endpoint of l_1^+ at infinity); otherwise, α_1 is defined as l_1^+ and α is updated to l_2^+. In general, right before the i-th iteration (i.e., right before l_i^+ is processed), a y-monotone rightward convex chain α is maintained and the lower endpoint of α is p_{i-1} (see Fig. 6). The general algorithm for the i-th iteration works as follows. By traversing on α from its lower endpoint p_{i-1}, we find the intersection q between α and l_i^+ (see Fig. 6). If q exists, then α_{i-1} is defined as the portion of α between p_{i-1} and q, and α is updated to the union of $\overline{p_i q}$ and $\alpha \setminus \alpha_{i-1}$. If q does not exist, then $\alpha_{i-1} = \alpha$ and $\alpha = l_i^+$. In either case, the new α is a y-monotone rightward convex chain with lower endpoint at p_i. We then proceed on the next iteration. The algorithm stops once l_n^+ is processed, after which F and all chains $\alpha_1, \alpha_2, \ldots, \alpha_n$ are obtained.

For the time analysis, observe that each i-th iteration takes time proportional to the number of edges of the chain α_{i-1} because we traverse α starting from p_{i-1}. As F is the edge-disjoint union of all chains α_i, $1 \leq i \leq n$, the total number of edges of all chains is equal to the number of edges of F, which is at most $2n-1$. Hence, the

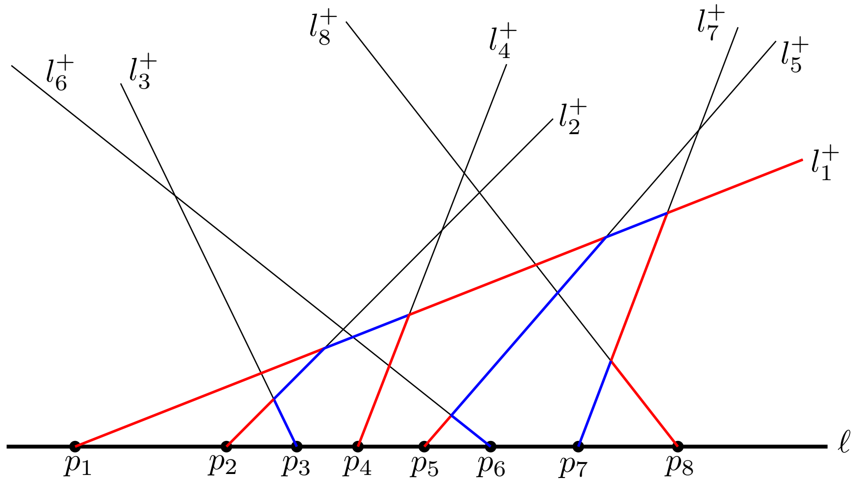

Figure 5: Illustrating the decomposition of F into y-monotone rightward convex chains α_i, $1 \leq i \leq n$: Each chain α_i is distinguished with the same color starting from the point p_i. For example, the blue chain starting from p_6 is α_6, which has three segments.

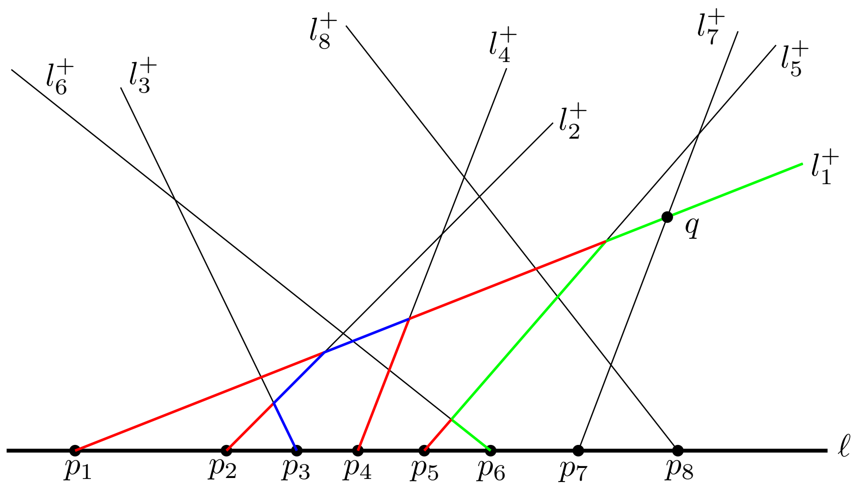

Figure 6: Illustrating the chain α (colored green) right before l_7^+ is processed. After l_7^+ is processed, α_6 becomes the portion of α between p_6 and q, and α is updated to $\overline{p_7q} \cup \overline{qp'_1}$, where p'_1 is the upper endpoint of l_1^+ at infinity.

time of the algorithm is $O(n)$.

Analogously, we can decompose the backward forest F' into n y-monotone leftward convex chains $\beta_1, \beta_2, \ldots, \beta_n$, such that F' is the edge-disjoint union of all these chains (see Fig. 7). For each i, the lower endpoint of β_i is p_i. Specifically, β_1 is the path from p_1 to its root in F'. For each $2 \leq i \leq n$, β_i is the sub-path from p_i to its tree root until the first vertex in the union of the chains $\beta_1, \beta_2, \ldots, \beta_{i-1}$. The forest F', along with the chains β_i, $1 \leq i \leq n$, can be computed in $O(n)$ time by an algorithm similar to the above for F.

Computing the cells C_i^+. We are now ready to compute the cells C_i^+, $1 \leq i \leq n$, using the convex chains of F and F' computed above. To this end, the following lemma is critical. Let ∂C_i^+ denote the boundary of C_i^+.

LEMMA 2.1. *For each $1 \leq i \leq n$, the boundary ∂C_i^+ can be identified as follows.*

- *For $1 \leq i \leq n-1$, if α_i and β_{i+1} intersect, say, at a point q, then ∂C_i^+ consists of the following three parts: $\overline{p_ip_{i+1}}$, the portion of α_i between p_i and q, and the portion of β_{i+1} between p_{i+1} and q (see Fig. 8); otherwise, $\partial C_i^+ = \overline{p_ip_{i+1}} \cup \alpha_i \cup \beta_{i+1}$.*
- *For $i = n$, $\partial C_i^+ = \overline{p_np_{n+1}} \cup \alpha_n$ (see Fig. 5).*
- *For $i = 0$, $\partial C_i^+ = \overline{p_0p_1} \cup \beta_1$ (see Fig. 7).*

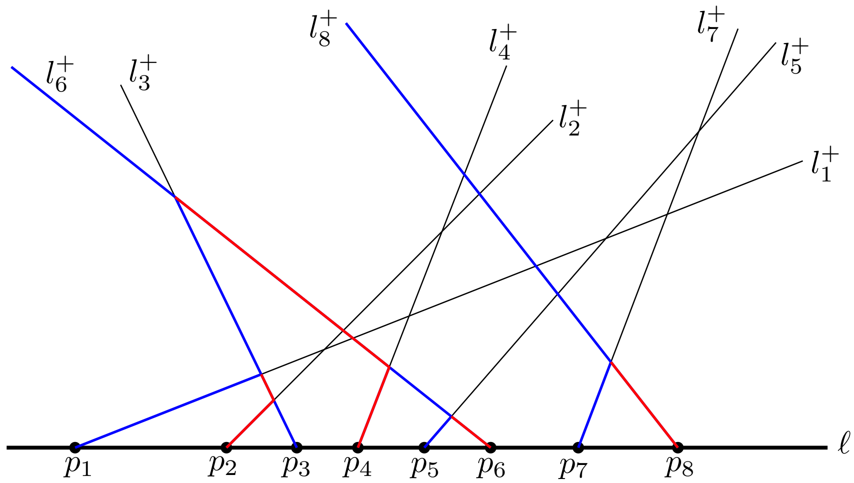

Figure 7: Illustrating the decomposition of F' into y-monotone leftward convex chains β_i, $1 \leq i \leq n$: Each chain β_i is distinguished with the same color starting from the point p_i. For example, the blue chain starting from p_7 is β_7, which has two segments.

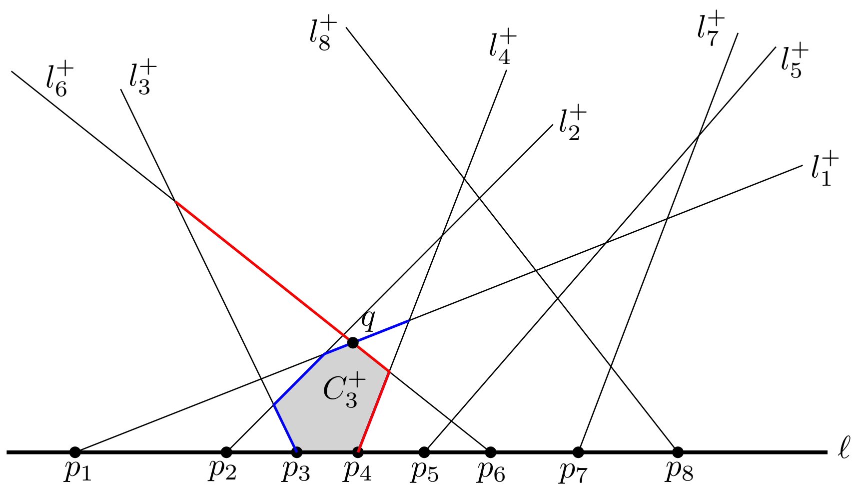

Figure 8: The blue chain is α_3 and the red chain is β_4, and they intersect at q. The grey region is C_3^+, whose boundary consists of $\overline{p_3p_4}$, the portion of α_3 between p_3 and q, and the portion of β_4 between p_4 and q, with $q = \alpha_3 \cap \beta_4$.

Proof. We only prove the general case $1 \leq i \leq n-1$, since the other two special cases can be proved analogously (and in an easier way).

In the following, unless otherwise stated, all points in question are above ℓ. Also, when we say a point p is to the right (resp., left) of a half-line l_i^+, it includes the case $p \in l_i^+$, i.e., the x-coordinate of p is larger than (resp., smaller than) or equal to that of p', where p' is the intersection between l_i^+ and the horizontal line through p.

We first have the following observation about C_i^+: a point p is in C_i^+ if and only if p is to the right of l_j^+ for all $j \leq i$ and is also to the left of l_j^+ for all $j \geq i+1$.

Recall that both α_i and β_{i+1} are y-monotone convex chains with their lower endpoints at ℓ. Let A_i refer to the region of the plane bounded by $\overline{p_ip_{i+1}}$, α_i, and the segment of l_{i+1}^+ on F (see Fig. 9). Note that if l_{i+1}^+ does not intersect α_i, then the upper endpoint of α_i is at infinity and the entire l_{i+1}^+ is on F, and thus A_i is unbounded. Since α_i is y-monotone, a point p is in A_i if and only if p is to the left of l_{i+1}^+ and also to the right of α_i (i.e., the horizontal line through p intersects α_i and p is to the right of the intersection). Further, by the definition of α_i, A_i consists of all points to the left of l_{i+1}^+ and to the right of the right envelope of the lines $\{l_1^+, l_2^+, \ldots, l_i^+\}$. Therefore, by the property of the right envelope, we obtain that a point p is in A_i if and only if p is to the left of l_{i+1}^+ and to the right of l_j^+ for all $j \leq i$. Due to the above observation on C_i^+, we obtain that $C_i^+ \subseteq A_i$.

Similarly, define B_{i+1} as the region of the plane bounded by $\overline{p_ip_{i+1}}$, β_{i+1}, and the segment of l_i^+ on F' (see

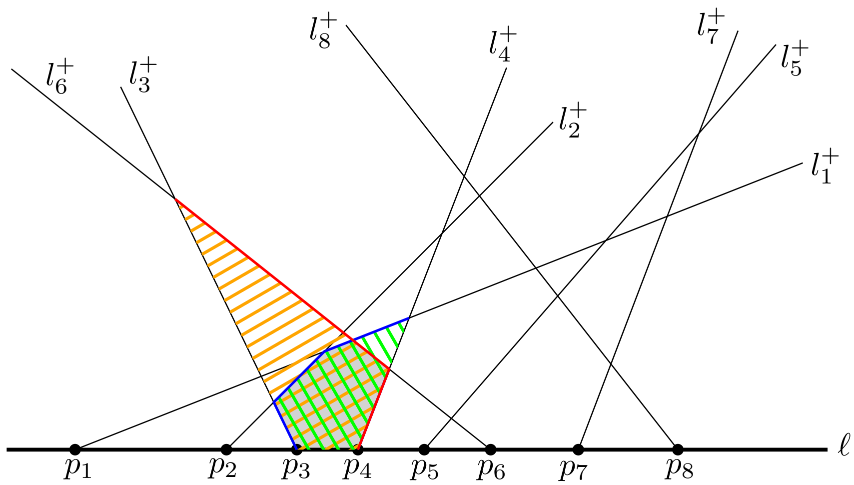

Figure 9: Illustrating the regions A_3 (green backslash region) and B_4 (orange slash region). The gray region, which is $A_3 \cap B_4$, is C_3^+. The blue chain is α_3 and the red chain is β_4.

Fig. 9). By an argument similar to the above, we have $C_i^+ \subseteq B_{i+1}$. Therefore, it follows that $C_i^+ = A_i \cap B_{i+1}$ (see Fig. 9).

Recall that the boundary ∂A_i consists of $\overline{p_i p_{i+1}}$, α_i, and the segment of l_{i+1}^+ on F. We call α_i and the segment of l_{i+1}^+ on F the *left and right boundaries* of A_i, respectively. Similarly, we call β_{i+1} and the segment of l_i^+ on F' the *right and left boundaries* of B_{i+1}, respectively. For C_i^+, recall that it is convex and has $\overline{p_i p_{i+1}}$ as an edge. We define its left and right boundaries similarly. Specifically, if C_i^+ is bounded, then the highest vertex of C_i^+ partitions $\partial C_i^+ \setminus \overline{p_i p_{i+1}}$ into two y-monotone convex chains, and the left one is called the *left boundary* of C_i^+ and the right one is called the *right boundary.* If C_i^+ is unbounded, then $\partial C_i \setminus \overline{p_i p_{i+1}}$ consists of two y-monotone convex chains going upwards to infinity, and the left one is called the *left boundary* of C_i^+ and the right one is called the *right boundary.* In either case, the left boundary of C_i^+ is y-monotone rightward convex and the right boundary of C_i^+ is y-monotone leftward convex.

Because $C_i^+ = A_i \cap B_{i+1}$, and A_i, B_{i+1}, and C_i^+ are all bounded by $\overline{p_i p_{i+1}}$ from below, the left boundary of C_i^+ belongs to the right envelope of the left boundary of A_i and the left boundary of B_{i+1}. Recall that if we move on the left boundary of A_i from p_i, then we first move on l_i^+ and then always turn rightwards. On the other hand, the left boundary of B_{i+1} is a segment of l_i^+ with p_i as an endpoint. This implies that the left boundary of C_i^+ must be a subset of the left boundary of A_i, i.e., α_i, because the left boundaries of A_i, B_{i+1}, and C_i^+ are all rightward convex. A symmetric argument shows that the right boundary of C_i^+ must be a subset of the right boundary of B_{i+1}, i.e., β_{i+1}. Therefore, if C_i^+ is closed, then the left boundary of ∂C_i^+ is the portion of α_i between p_i and q, and the right boundary of ∂C_i^+ is the portion of β_{i+1} between p_{i+1} and q, where q is the intersection of α_i and β_{i+1}. If C_i^+ is open, then the left boundary of C_i^+ is α_i and the right boundary of C_i^+ is β_{i+1}. This proves the lemma. ∎

Based on Lemma 2.1, the cells C_i^+, $1 \le i \le n$, can be easily computed. First of all, the two cells C_0^+ and C_n^+ are already available because both α_n and β_1 have been computed. For each $1 \le i \le n-1$, we can compute C_i^+ using the two convex chains α_i and β_{i+1} as follows. By Lemma 2.1, it suffices to find the intersection q between α_i and β_{i+1}, or determine that such an intersection does not exist. This can be done in $O(|\alpha_i| + |\beta_{i+1}|)$ time by a straightforward sweeping algorithm similar to that for merging two sorted lists. Indeed, starting from ℓ, we sweep a horizontal line h upwards. During the sweeping, we maintain the edges of α_i and β_{i+1} intersecting h. An event happens if h encounters a vertex of α_i or β_{i+1}. Consider an event at a vertex $a \in \alpha_i$, i.e., $a \in h$. Let a' be the next vertex of α_i after a, i.e., $\overline{aa'}$ is the edge of α_i right above h (see Fig. 10). Let $\overline{bb'}$ be the edge of β_{i+1} currently intersecting h such that b' is the upper vertex of the edge. To process the event, we first check whether $\overline{aa'}$ and $\overline{bb'}$ intersect. If yes, then their intersection is q and we stop the algorithm. Otherwise, we proceed on the next event, which is the lower point of a' and b'. If q is not found after all events are processed, then α_i and β_{i+1} do not intersect and we stop the algorithm. It is easy to see that the algorithm runs in $O(|\alpha_i| + |\beta_{i+1}|)$ time.

Therefore, computing all cells C_i^+ for all $i = 0, 1, \ldots, n$ takes time proportional to $\sum_{i=1}^n |\alpha_i| + \sum_{i=1}^n |\beta_i|$,

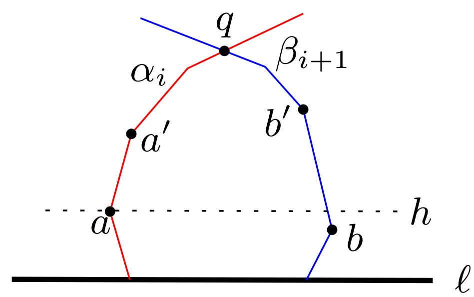

Figure 10: Processing an event at a.

which is $O(n)$. Consequently, $Z^+(\ell)$, which is the disjoint union of all cells C_i^+, for all $i = 0, 1, \ldots, n$, is obtained. Note that since we already have the sorted list $p_1, p_2, \ldots, p_n$, we could also easily connect cells C_i^+ together from left to right on ℓ.

Again, we can use a similar algorithm to compute $Z^-(\ell)$ in $O(n)$ time. Finally, the zone $Z(\ell)$ is simply the disjoint union of $Z^+(\ell)$ and $Z^-(\ell)$. The following theorem summarizes our result.

THEOREM 2.1. *Given a set L of n lines in the plane and another line ℓ, the zone $Z(\ell)$ in the arrangement of L can be computed in $O(n \log n)$ time. If the sorted list of the intersections between ℓ and all lines of L is known, then $Z(\ell)$ can be computed in $O(n)$ time.*

Dealing with degeneracies. There are two degenerate cases: (1) L has horizontal lines; (2) ℓ contains intersections of lines of L.

To handle the first case, without loss of generality, we assume that L has horizontal lines above ℓ, and among those, let ℓ' be the lowest one. We first compute the "upper zone" $Z^+(\ell)$ as usual without considering the horizontal lines of L. Then, for each cell C_i^+ of $Z^+(\ell)$, we simply cut it along ℓ' (alternatively, we could easily incorporate this cut operation into our sweeping algorithm for computing C_i^+). The union of all cells C_i^+ after the cut is the upper zone $Z^+(\ell)$ of L including all horizontal lines. The total time of the algorithm does not change asymptotically.

To handle the second case, what really matters is the sorted list $l_1, l_2, \ldots,$ of L, which is used in our forest decomposition algorithm. If two or more lines of L have a common intersection on ℓ, then we break the tie by further comparing their slopes: a line l is placed in the sorted list in the front of another line l' if the half-line of l above ℓ is left of that of l'. After having the sorted list of L, we can run exactly the same algorithm as before.

Remark. Our algorithm also provides a simple proof for the combinatorial size of the zone $Z(\ell)$. If L has a horizontal line, a slight rotation of it only increases the complexity of $Z(\ell)$. Hence, it suffices to assume that L does not have any horizontal line. According to our algorithm, each edge of $Z^+(\ell)$ lies on an edge of one of the two forests F and F'. As discussed before, each forest has at most $2n - 1$ edges (even in the second degenerate case). Hence, the total number of edges of $Z^+(\ell)$ is at most $4n - 2$. Therefore, the total number of edges of the zone $Z(\ell)$ is at most $8n - 4$, which matches the bounds obtained in [6,10,12].

References

[1] P.K. Agarwal. Partitioning arrangements of lines II: Applications. *Discrete and Computational Geometry*, 5:533–573, 1990.

[2] P.K. Agarwal, J. Matoušek, and O. Schwarzkopf. Computing many faces in arrangements of lines and segments. *SIAM Journal on Computing*, 27:491–505, 1998.

[3] P.K. Agarwal and M. Sharir. Arrangements and their applications, in *Handbook of Computational Geometry,* J. Sack and J. Urrutia (eds.), pages 49–119. Elsevier, Amsterdam, The Netherlands, 2000.

[4] P. Alevizos, J.-D. Boissonnat, and F.P. Preparata. An optimal algorithm for the boundary of a cell in a union of rays. *Algorithmica*, 5:573–590, 1990.

[5] M.W. Bern, D. Eppstein, P.E. Plassmann, and F.F. Yao. Horizon theorems for lines and polygons. *Discrete and Computational Geometry: Papers from the DIMACS Special Year*, 6:45–66, 1991.

[6] B. Chazelle, L.J. Guibas, and D.T. Lee. The power of geometric duality. *BIT*, 25:76–90, 1985.

[7] H. Edelsbrunner. *Algorithmis in Combinatorial Geometry.* Springer-Verlag, Heidelberg, Germany, 1987.

[8] H. Edelsbrunner and L. Guibas. Topologically sweeping an arrangement. *Journal of Computer and System Sciences*, 38(1):165–194, 1989.

[9] H. Edelsbrunner, L. Guibas, J. Hershberger, R. Seidel, M. Sharir, J. Snoeyink, and E. Welzl. Implicitly representing arrangements of lines or segments. *Discrete and Computational Geometry*, 4:433–466, 1989.

[10] H. Edelsbrunner, L. Guibas, J. Pach, R. Pollack, R. Seidel, and M. Sharir. Arrangements of curves in the plane topology, combinatorics, and algorithms. *Theoretical Computer Science*, 92(2):319–336, 1992.

[11] H. Edelsbrunner and E.P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics*, 9:66–104, 1990.

[12] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15:341–363, 1986.

[13] H. Edelsbrunner, R. Seidel, and M. Sharir. On the zone theorem for hyperplane arrangements. *SIAM Journal on Computing*, 22:418–429, 1993.

[14] D. Halperin and M. Sharir. Arrangements, in *Handbook of Discrete and Computational Geometry,* C.D. Tóth, J. O'Rourke, and J.E. Goodman (eds.), pages 723–762. CRC Press, 3rd edition, 2017.

[15] M. Sharir and P.K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications.* Cambridge University Press, 1995.