A Survey on Arrangement of Lines

Md. Ehtesamul Haque (#100605034P) and Masud Hasan
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh

April 15, 2007

Abstract

Arrangement of lines is the subdivision of a plane by a finite set of lines. Arrangement is an important structure which can aid in solving many problems in theoretical computer science. Arrangement of lines also has important application in robot motion planning and computer graphics. In this paper, we discuss about combinatorics of arrangement, substructures of arrangement and algorithm to find an arrangement.

Key words: Arrangement, Zone theorem.

1 Introduction

An *Arrangement of lines*, $\mathcal{A}(L)$ is the division of a plane by a finite set of lines, L. Arrangements[1] occur frequently in many computational geometry problems. Many of the theoretical computer science problems have arrangement underlying their problem specification. Arrangement has important application in robot motion planning through visibility graph [9, 11]. It has also important application in computer graphics specially in hidden surface removal [10]. Arrangement of more complex geometric elements such as arrangement of curves and surfaces is also important as it has application in molecular modeling [3].

Arrangement has been studied since late $60'$s and has a rich history of work. Initial idea of arrangements has been conceived by Grünbaum [7, 6, 5]. Later, Edelsbrunner *et. al.* introduced more complex arrangement of hyperplanes [3]. He also summarized the works on arrangements up to 1987 [2]. A comprehensive discussion on arrangement can be found in [8]. In this paper, we mainly summarize the chapter by Halperin in [8]. However discussion in [8] is on general dimension arrangement *i. e.* arrangement of hyperplanes whereas we concentrate on arrangement of lines here. In section 2, we discuss about some basic terms related to lines and arrangements. Section 3 deals with the complexity of an arrangement. Some subtle and critical substructures are discussed in section 4. In section 5, we discuss algorithms to determine arrangement. Finally, section 6 concludes this paper.

[1]In this paper, we use arrangement to mean arrangement of lines unless explicitly specified otherwise.

2 Preliminaries

A *line* on a plane is an infinite set of points which are equidistant from two fixed points. Thus, a line extends to infinity. Let, L be a set of n lines on a plane. The set of lines L, will split the plane into some convex regions with some edges and vertices. This division is called arrangement, $\mathcal{A}(L)$, induced by a set of lines L. A *vertex* of an arrangement, $\mathcal{A}(L)$, is an intersection point of two lines l_1 and l_2, where $l_1, l_2 \in L$. An *edge* of an arrangement, $\mathcal{A}(L)$, is the line segment between two vertices v_1 and v_2 which contains no other vertex, where v_1, v_2 are two vertices of arrangement, $\mathcal{A}(L)$. Moreover, a line segment that extends from a vertex v_1 to infinity and does not contain any other vertex except v_1 is also called an edge. Such an edge is called *unbounded edge.* Thus, an edge whose both endpoints are vertices of arrangement, $\mathcal{A}(L)$, is called a *bounded edge.* A *face* of an arrangement, $\mathcal{A}(L)$, is a maximal region created by the set of lines, L, which does not contain any vertex or edge of arrangement, $\mathcal{A}(L)$, in its interior. Note that similar to edges, faces can also be unbounded. A face f, of arrangement $\mathcal{A}(L)$, is called *bounded face* if its boundary is a closed polygon formed by the edges of $\mathcal{A}(L)$. The union of bounded faces in an arrangement, $\mathcal{A}(L)$, is called the *envelope of arrangement*, $\mathcal{A}(L)$. *Complexity of the envelope of an arrangement* is the number of edges in the envelope. Figure 1 is an arrangement of 6 lines with 13 vertices, 32 edges

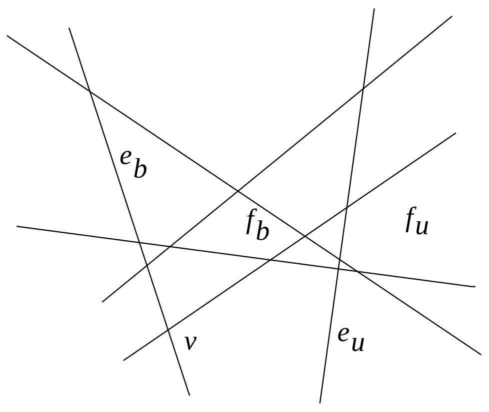

Figure 1: An arrangement of lines

and 20 faces, where v is a vertex, e_b is a bounded edge, e_u is an unbounded edge, f_b is an bounded face and f_u is an unbounded face.

An arrangement, $\mathcal{A}(L)$ is called simple if its line set, L, has two properties,

(i) Every two lines of L meet at exactly one point.

(ii) No three lines of L meet at a single point.

Figure 2(a) depicts a simple arrangement of 5 lines and 2(b) shows a nonsimple arrangement of 4 lines. An edge, e is said to *incident to a face* f, if e is part of the boundary of the face, f. It should be noted that a single edge will be incident to exactly two faces. The number of edges that are incident to a face, f, is called the *complexity of face*, f. Thus, if $\mathcal{F}$ be the set of faces in an arrangement, $\mathcal{A}(L)$, n_e is the number of edges in arrangement, $\mathcal{A}(L)$, and $\mathcal{C}(f)$ be the complexity

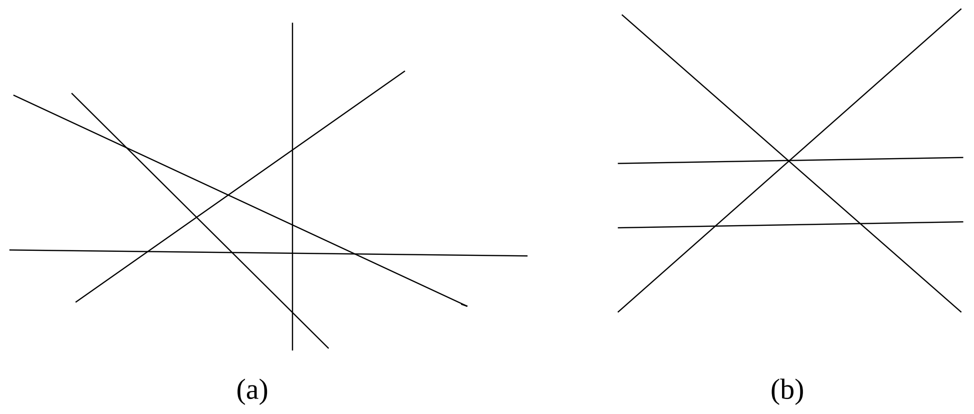

Figure 2: (a)A simple arrangement of 5 lines and (b) An arrangement of 4 lines that is not simple.

of face f, then

$$\sum_{f \in \mathcal{F}} \mathcal{C}(f) = 2 \times n_e.$$

If $\mathcal{A}(L)$ is an arrangement induced by a set of lines L, then The *zone of a line* l, in $\mathcal{A}(L)$, is the set of faces intersected by l and is denoted by $Z_{\mathcal{A}}(l)$ or simply $Z(l)$ when arrangement is clear from the context. The *complexity of the zone* $Z_{\mathcal{A}}(l)$, is the total complexity of the faces in that zone, $Z_{\mathcal{A}}(l)$. So, $\mathcal{C}(f)$ be the complexity of face f, complexity of $Z_{\mathcal{A}}(l) = \sum_{f \in Z_{\mathcal{A}}(l)} \mathcal{C}(f)$.

3 Complexity of Arrangements

Presence of some lines on a plane induces some vertices, edges and faces. The complexity of an arrangement is expressed by three numbers. The numbers that are used to express the complexity of an arrangement are,

- number of vertices,
- number of edges and
- number of faces.

It is easier to measure the complexity of a simple arrangement. So, we concentrate on complexity of a simple arrangement first. To measure the complexity of a simple arrangement, we have the following theorem,

Theorem 3.1 *If L is a set of n lines and $\mathcal{A}(L)$ is a simple arrangement induced by L, then*

(i) the number of vertices in $\mathcal{A}(L)$ is $\frac{n(n-1)}{2}$,

(ii) the number of edges in $\mathcal{A}(L)$ is n^2 and

(iii) the number of faces in $\mathcal{A}(L)$ is $\frac{n^2}{2} + \frac{n}{2} + 1$.

Proof *(i)* The number of vertices in an arrangement is simply the number of intersection points of pairs of lines in L. Now, in a simple arrangement every pair of lines in L meet at a unique point. Hence, The number of vertices in a simple arrangement is equal to the number of pairs of lines in L which is $\binom{n}{2}$. Thus number of vertices in a simple arrangement is $\binom{n}{2} = \frac{n(n-1)}{2}$.

(ii) We use induction method to establish the number of edges in a simple arrangement. First, when $n = 1$, there is only one line in the arrangement and thus only $1 = 1^2$ edge.

Now, assume in a simple arrangement of $n-1$ lines, there are $(n-1)^2$ edges. When we add the nth line, it will intersect previous all $n-1$ edges creating $n-1$ more edges. Moreover, the nth line itself will create n new edges as it has intersected at $n-1$ points. Thus,

$$\text{total number of edges} = (n-1)^2 + (n-1) + n = n^2 - 2n + 1 + 2n - 1 = n^2.$$

(iii) Induction method can be used to determine the number of faces in a simple arrangement. When $n = 1$, The face in the arrangement is 2 as there is a face in either side of the line. Again, $\frac{n^2}{2} + \frac{n}{2} + 1 = \frac{1}{2} + \frac{1}{2} + 1 = 2$. Thus number of faces is $\frac{n^2}{2} + \frac{n}{2} + 1$ when $n = 1$.

Now assume there are $\frac{(n-1)^2}{2} + \frac{n-1}{2} + 1$ faces in a simple arrangement of $n-1$ lines. When the nth line is added, it will be divided in to n edges and each of these edges will split different faces. So, n new faces will be added. So,

$$\text{total number of faces} = (\frac{(n-1)^2}{2} + \frac{n-1}{2} + 1) + n = \frac{n^2}{2} + \frac{n}{2} + 1$$

This concludes proof of theorem 3.1. ∎

Number of vertices in a simple arrangement is obviously higher than that of nonsimple arrangement when number of lines in the arrangements is same. This is because, in a simple arrangement, every pairs of lines creates an unique intersection point which is not the case in a nonsimple arrangement. So, Number of vertices in a nonsimple arrangement is lower than it's simple counterpart. Lower number of vertices leads to lower number of edges and faces. So, for general arrangements we have the following corollary,

Corollary 3.2 *If L is a set of lines and $\mathcal{A}(L)$ is the arrangement induced by L, then*

(i) the number of vertices in $\mathcal{A}(L)$ is at most $\frac{n(n-1)}{2}$,

(ii) the number of edges in $\mathcal{A}(L)$ is at most n^2 and

(iii) the number of faces in $\mathcal{A}(L)$ is at most $\frac{n^2}{2} + \frac{n}{2} + 1$. ∎

4 Substructures in Arrangements

A substructure of an arrangement is a portion of an arrangement that may be important in solving a problem. Sometimes the total complexity of an arrangement does not needed to be handled, rather a subset of an arrangement may be enough to solve a problem. In this section we discuss about three important substructures of arrangements.

4.1 Faces in an Arrangement

Faces alone do not sound much important. But, the implication of faces in arrangement is very important. Complexity of faces dictates the strategy and runtime of many arrangement algorithms that have significant impact in the applications where arrangement is applicable. Many properties, such as convexity of faces in an arrangement, simplifies many analysis and strategy in solving problems relating arrangements. Here we focus on two important properties of faces in arrangement namely, convexity and complexity of faces.

Every faces in arrangement are convex. Although faces may be bounded or unbounded in an arrangement there is no non-convex face in an arrangement. We can prove it using following theorem,

Theorem 4.1 *Every face f, in an arrangement $\mathcal{A}(L)$, induced by the set of lines L, is convex.*

Proof For a contradiction, let f' is a non-convex face of $\mathcal{A}(L)$ and u, v be two points contained by the face f', such that their connecting line uv is not totally contained by f'. Then there is some lines crossing the segment uv. Let, l be such a line as shown in figure 3. Then u and v are in

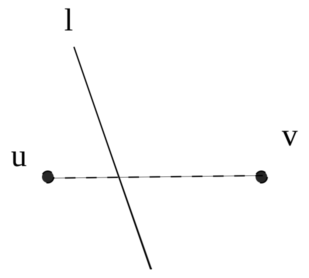

Figure 3: Two points in a non-convex face.

opposite side of line l. But being a line, l extends to infinity. So, there is no way for the two points u,v, which are in opposite side of a line, to be in a single face. So, there is no non-convex face in $\mathcal{A}(L)$. ▮

The complexity of a face is the number of edges in the boundary of that face. Since every face is convex, no line can create two edges of a single face. So, the complexity of a single face can be at most n, in an arrangement $\mathcal{A}(L)$ which is induced by a set of n lines, L. Again, given n it is always possible to build an n-gon which will ensure presence of a face with worst complexity. Figure 4 shows an arrangement of 5 lines where a 5-gon is present. So, the complexity of a single face is $O(n)$.

4.2 Zone of a Line

Zone of a line in an arrangement is important since many of the algorithms to build arrangements depend on zone of lines. So, performance of that algorithms also is dictated by complexities of the zones of lines. To measure the complexity of a zone, the following theorem is important, which is widely known as zone theorem.

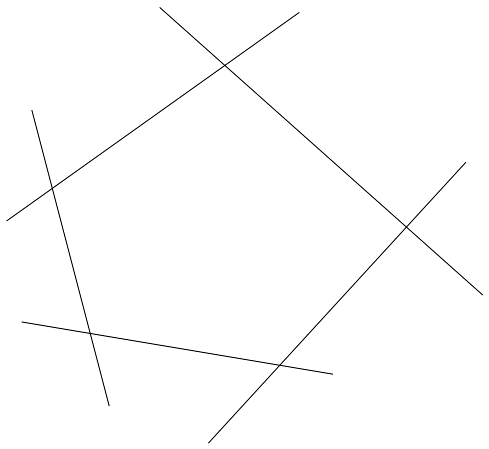

Figure 4: An arrangement of 5 lines with a 5-gon.

Theorem 4.2 *Let L be a set of n lines and $\mathcal{A}(L)$ be the arrangement induced by L. If l is another arbitrary line and $Z_{\mathcal{A}}(l)$ is the zone of line l, in the arrangement $\mathcal{A}(L)$, then the complexity of the zone $Z_{\mathcal{A}}(l)$, is $O(n)$.*

Proof We assume the arrangement is simple. Since we are proving upper bound assuming the arrangement to be simple does no harm as we have already seen from section 3 that simple arrangement contains most number of faces. We also assume that the line l is horizontal. If l is not horizontal, the coordinate axis can be rotated to make it horizontal. we prove theorem 4.2 by induction.

When $n = 1$, number of lines is one. So, there are two faces and any new line can only intersect these $2 = n + 1$ faces. Now, we assume the hypothesis is true for an arrangement of $n - 1$ lines. Since l is horizontal and we assumed a simple arrangement, no other line is horizontal. So, it is possible to define all edges as left bounding or right bounding edges with respect to faces. An edge e, is a *left bounding edge* of a face f, if e is incident to f and there is a horizontal line h which intersect e at a point x, such that immediate right point of x is in f. Similarly a *right bounding edge* can be defined. Now, we focus on proving the bound on left bounding edge to be $\leq 3n$. Similar bound on right bounding edges can be combined with that of left bounding edges to prove the upper bound on zone complexity. Since we assumed our hypothesis is true for $n - 1$ lines, number of left bounding edges in an arrangement of $n - 1$ lines is $l_{n-1} \leq 3(n - 1)$. Now to prove this bound in an arrangement of n lines, we will remove one line from $\mathcal{A}(L)$ and show that adding back this line will increase the number of left bounding edge by at most three.

Given $\mathcal{A}(L)$, an arrangement of n lines and another line l, we select a line, $r \in L$ which intersects l at the rightmost point among all lines in L. We need to prove that adding r back will not create more than 3 new left bounding edges. Putting r back may increase the number of left bounding edges in two ways, r itself may create new left bounding edges and r may intersect other left bounding edges.

Figure 5 shows a condition created when r is put back onto the arrangement. We now show that r it self may create at most one new left edge. Let u be the point where r first intersects with any line above l and v be the point where r first meets with any line below l. Clearly, uv is a left

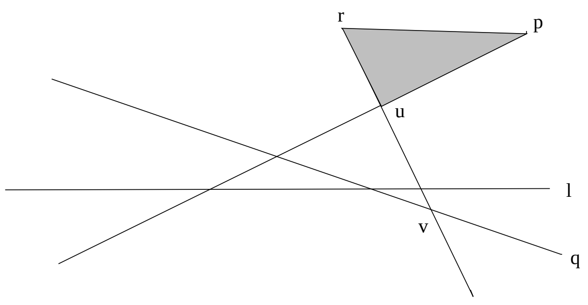

Figure 5: The rightmost intersection line of l.

bounding edge of the unbounded face right to r. So, r it self creates one new left bounding edge. Now, consider the region R (colored gray in figure 5), between r and p above u, this region cannot be in the zone of line l in $\mathcal{A}(L)$. Since R is bounded on the left by r and on the right by p, there is no way l can intersect R. Thus the portion of r above u cannot carate any left bounding edge. Similar argument can show that the portion of r below v cannot form any left bounding edge. So, r itself can create at most one new left bounding edge.

Now, we prove that r may increase the number of old left bounding edges by at most two. We observe that r may increase the number of left bounding edge by intersecting an old left bounding edge e, if e is an edge of the rightmost unbounded face of the zone of line l. The reason behind

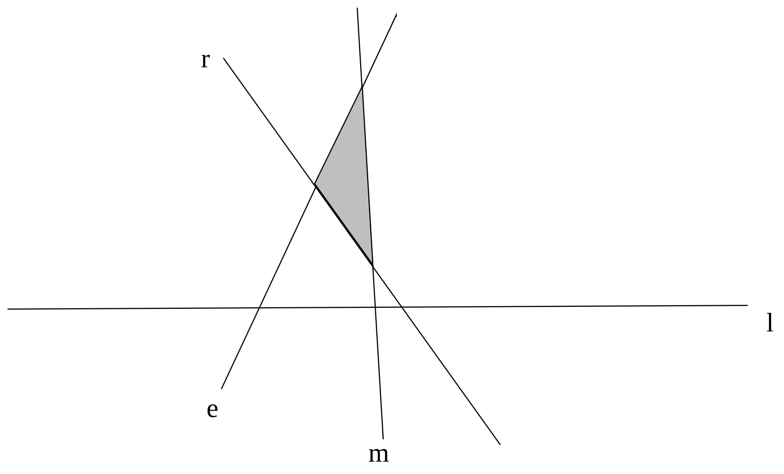

Figure 6: No increase of left bounding edge.

that is if r splits any left bounding edge e (refer to figure 6), of a bounded face then there will be at least one edge m that intersects r at a point right to the intersecting point of e and r. So, the region bounded by r, e and m (colored gray in figure 6) cannot be in the zone of line l as it has no way to incident to l. Thus r can only split an old left bounding edge e, to create two new left bounding edge if e is a left bounding edge of the rightmost unbounded face in the zone of the line l. Figure 7 shows that case where previous left bounding edge uw is split into two new left bounding edges uv and vw, by r. According to theorem 4.1 every face of the arrangement is convex. So, the rightmost unbounded face also is convex and being a straight line, r can intersect that face at at

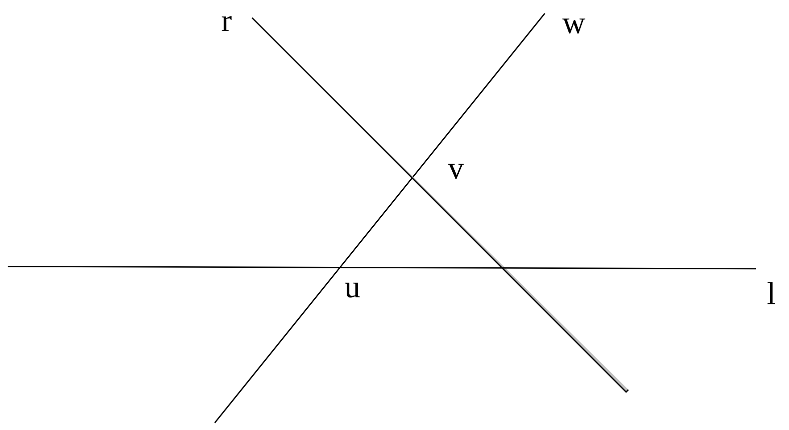

Figure 7: Increase of left bounding edge.

most two points. Thus r can split two old left bounding edges causing an increase of two in the number of left bounding edges. So, total number of increase in left bounding edge number is three as r itself can create one more left bounding edge. So, by induction method number left bounding edges in an arrangement of n lines, $l_n \leq 3n$.

The number of right bounding edges can be bound to be $3n$, similarly. Combining these two bounds, we can say that the complexity of $Z_{\mathcal{A}}(l)$ is indeed $O(n)$. ▮

4.3 Envelope of an Arrangement

Envelope of an arrangement is an important substructure of arrangements. Determining envelope of an arrangement has application in linear programming. Moreover determining upper (lower) envelope of an arrangement is equivalent to determining lower (upper) convex hull in dual space. In this subsection, we document a lemma related to complexity of the envelope of an arrangement due to J. Urrutia. We specify the lemma without proof. Proof of lemma 4.3 can be found in [4].

Lemma 4.3 *Let $\mathcal{A}(L)$ be the arrangement induced by a set of n lines L, and P be the envelope of $\mathcal{A}(L)$, then P contains at most $3.5n$ edges.* ▮

5 Algorithm to Construct an Arrangement

In this section, we focus on algorithmic approach to build an arrangement from the set of lines that is inducing the arrangement. An important question comes in mind about what should be the input and output of an algorithm that constructs an arrangement. The input is the set of lines. As a line can be expressed using two numbers, n pairs of numbers suffice as input. Output is a little bit tricky as it may be dependent on the problem that is solved by arrangement. Thus special data structures are used to denote arrangement. So, we discuss about data structure that may be used to store an arrangement.

5.1 Data Structure for Arrangement

The choice of data structure is primarily dictated by the application that are being tackled using arrangement. Here we discuss about widely used $O(n^2)$ data structure, *Doubly-Connected Edge List*. In an arrangement every edge is incident to two faces. So, each edge is considered as two half edges each one incident to one face. These two half edges are called *twin* to each other. Each half edge is oriented in a direction so that if any one walks along a half edge the incident face is at the left of the walker. The vertex at the start of the half edge, along the direction of that half edge, is called the *origin* of that half edge. It should be noted that the destination of a half edge is the origin of the twin half edge. Moreover every vertex and face has also one record in this data structure. So, there are three types of records in doubly-connected edge list data structure,

- **vertex**: vertex record for the vertex v stores the coordinate of v. It also stores an arbitrary half edge $\overrightarrow{e}$ whose origin is v.
- **face**: face record for a face f, stores two components named outercomponent and innercomponent. Outercomponent is a half edge of the outer boundary of face f and innerboundary is a half edge of the inner boundary of face f.
- **half edge**: Record for a half edge $\overrightarrow{e}$ stores origin, twin, incident face, next half edge and previous half edge. Origin is a pointer to the record for the vertex which is the origin of $\overrightarrow{e}$. Twin is a pointer to the record of the twin half edge of $\overrightarrow{e}$. Incident face stores a pointer to the face record to which $\overrightarrow{e}$ is incident. Next half edge stores a pointer to the half edge that comes after $\overrightarrow{e}$ while traversing boundary of incident face of $\overrightarrow{e}$. Similarly previous face is stored.

5.2 Incremental Algorithm to Build an Arrangement

In this subsection, we illustrate an incremental algorithm to build a doubly connected edge list of an Arrangement. In this incremental algorithm, lines are added one by one and faces created by lines added so far are split by the new line. A problem occurs relating unbounded face and initial condition. There is no face initially. So, something special should be done. More over unbounded faces are hard to split as they cannot be traversed fully. The problem is solved using a bounding box that contains every vertices of the arrangement. Initially there is only one face bounded by the bounding box. Details of the algorithm is described below.

Let, L be a set of n lines such that $L = \{l_1, l_2, \ldots, l_n\}$. A bounding box $\mathcal{D}(L)$, is calculated from the whole set of lines, L. Let, $\mathcal{A}_i$ denotes the subdivision of the plane induced by bounding box $\mathcal{D}(L)$, and part of $\mathcal{A}(\{l_1, l_2, \ldots, l_i\})$ that is inside $\mathcal{D}(L)$. While adding line l_i, we must split the faces of $\mathcal{A}_{i-1}$ that are intersected by l_i. The splitting starts with determining the left most face that is intersected by l_i. When the line line l_i enters into face f, half edges of the boundary of f is traversed to determine the edge e where line l_i is leaving face f. Once leaving edge e is found, face f is split into two face and half edge e is also split. Next intersecting face is founding twin of e as next intersecting face is incident to twin of e.

Figure 8 shows splitting of face f by a line l. The entering edge is already split while handling previous face. Now, we create one new record for face, create two new half edge record created by

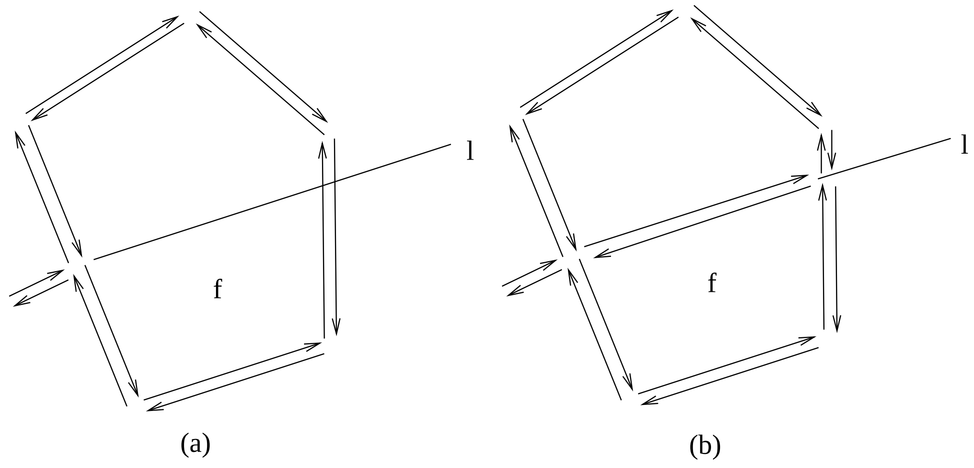

Figure 8: Splitting of a face.

l itself inside the face f. Finally, the edge e is split where l leaves the face. This splitting continues until an unbounded face that is face outside the bounding box $\mathcal{D}(L)$ is reached. The algorithm can be summarized as follows,

```
Algorithm BuildArrangement, given the line set $L$
input: a set of $n$ lines, $L$
output: A doubly connected edge list representing arrangement induced by $L$, bounded by a box.
 1: Compute a bounding box $\mathcal{D}(L)$ that contains all the vertices of the arrangement induced by $L$.
 2: Build a doubly connected edge list recording vertices, edges and face created by $\mathcal{D}(L)$.
 3: for $i = 1$ to $n$ do
 4:     Find the left most edge $\overrightarrow{e}$ where line $l_i$ intersects with $\mathcal{A}_{i-1}$.
 5:     let $f$ be face incident to $\overrightarrow{e}$.
 6:     while $f$ is inside the bounding box $\mathcal{D}(L)$ do
 7:         Split f.
 8:         $f \leftarrow$ next intersected face by $l_i$.
 9:     end while
10: end for
```

The running time of this algorithm is $O(n^2)$. This can also be verified as, step 1 can be done in $O(n^2)$ time by determining all intersection points from L and taking a box having lines through the uppermost, leftmost, bottommost and rightmost points. Step 2 can be done in constant time. Step 4 takes $O(i)$ as left most line of the bounding box $\mathcal{D}(L)$, can be divided up to i pieces. Step 5 and 8 takes $O(1)$ time. Step 7 depends on the complexity of face f, which we know to $O(i)$ from subsection 4.1. But from theorem 4.2, we know step 6 and 7 run in time $O(i)$. So, total time needed for steps 3-10, $\sum_{i=1}^{n} O(i) = O(n^2)$. Thus total running time of algorithm BuildArrangement is $O(n^2) + O(1) + O(n^2) = O(n^2)$. More over the storage requirement of this algorithm is also quadratic, as output size is it self quadratic implied by corollary 3.2. So we can state the following theorem,

Theorem 5.1 *The doubly connected edge list for an arrangement induced by a set of n lines can be built in $O(n^2)$ time and storage.*

The proof of this theorem is obvious from above mentioned discussion. Since the output size itself is quadratic this algorithm is quite efficient in terms of running time. However, Edelsbrunner and Guibas have given an algorithm using topological sweep that works with linear storage maintaining quadratic runtime [1].

6 Conclusion

In this paper, we presented comprehensive discussion on arrangement of lines that might be important in the applications where arrangement is used to tackle critical problems. We have determined the size and complexity of a simple arrangement of lines. From that we have also given an upper bound on complexity of general arrangements.

Substructures of arrangements has many important implication on numerous problems. Envelope of an arrangement is attracting researchers attention since its introduction. We have discussed about three substructures of arrangements. We have documented a proof of the widely known theorem named zone theorem. We also have presented an incremental algorithm to build the arrangement in a special type of data structure called doubly connected edge list. We also discussed the basics of doubly connected edge list.

Arrangement can also be extended to higher dimensions like 3-dimension or d-dimension. Higher dimensional arrangement like arrangement of planes, is more challenging and requires subtle geometric understanding to deal with. Lack of Presence of simple algorithm to deal with higher dimensional arrangement makes them excellent choice to work with.

References

[1] Edelsbrunner, H. and Guibas, L. J., *Topologically sweeping an arrangement*, J. Comput. Syst. Sci., 38: 165-194, 1989. Corrigendum in 42:249-251, 1991.

[2] Edelsbrunner, H., *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, 1987.

[3] Edelsbrunner, H., O'Rourke, J. and Seidel, R., *Constructing arrangements of lines and hyperplanes with applications*, SIAM J. Comput., 15:341-363, 1986.

[4] Eu, D., Guévremont, E. and Toussaint, G. T., *On Envelopes of Arrangements of Lines*, J. Algorithms, 21(1):111-148, 1996.

[5] Grünbaum, B., *Arrangements and Spreads*, Regional Conf. Ser. Math., Amer. Math. Soc., Providence, RI, 1972.

[6] Grünbaum, B., *Arrangements of hyperplanes*, Congr. Numer., 3:41-106, 1971.

[7] Grünbaum, B., *Convex Polytopes*, Wiley, New York, 1967.

[8] Halperin, D., *Arrangements*, in Goodman, J. E. and O'Rourke, J., eds., *Handbook of Discrete and Computational Geometry*, CRC Press LLC, Boca Raton, Chapter 21:389-412, 1997.

[9] Halperin, D. and Sharir, M., *Arrangements and their applications in robotics: Recent developments.* In Goldberg, K., Halperin, D., Latombe, J. C. and Wilson, R., eds., *Algorithmic Foundations of Robotic.*, A. K. Peters, Boston, MA, 1995.

[10] McKenna, M., *Worst-case optimal hidden-surface removal algorithm*, ACM Trnas. Graph, 6:19-28, 1987.

[11] O'Rourke, J., *Art Gallery Theorems and Algorithms*, Oxford University Press, New York, 1987.