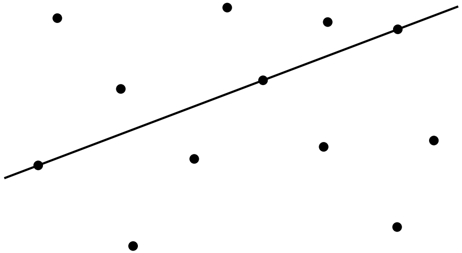
Arrangements and Duality

Computational Geometry

Lecture 9: Arrangements and Duality

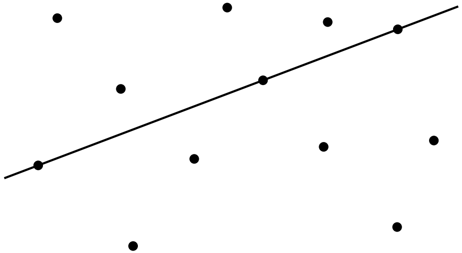
Three Points on a Line

Question: In a set of n points, are there 3 points on a line?



Three Points on a Line

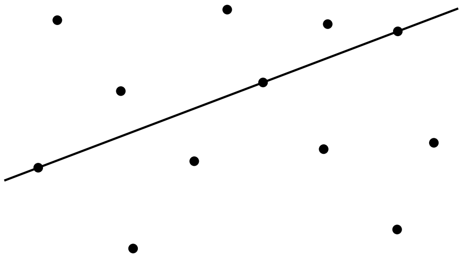
Question: In a set of n points, are there 3 points on a line?



Naive algorithm: tests all triples in $O(n^3)$ time

Three Points on a Line

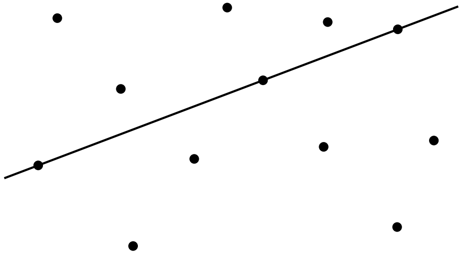
Question: In a set of n points, are there 3 points on a line?



Naive algorithm: tests all triples in $O(n^3)$ time
Faster algorithm: uses duality and arrangements

Three Points on a Line

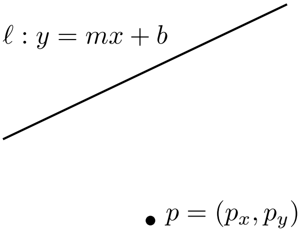
Question: In a set of n points, are there 3 points on a line?



Naive algorithm: tests all triples in $O(n^3)$ time
Faster algorithm: uses duality and arrangements

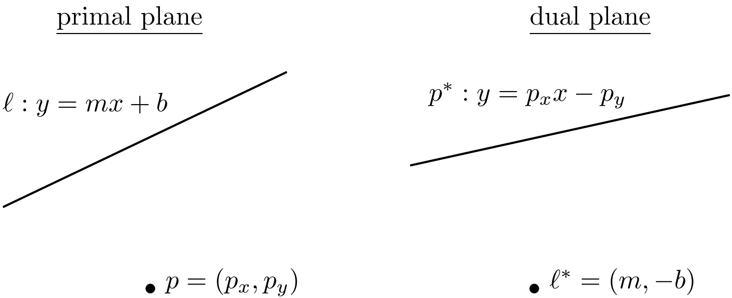
Note: other motivation in chapter 8 of the book

Duality



Note:

Duality

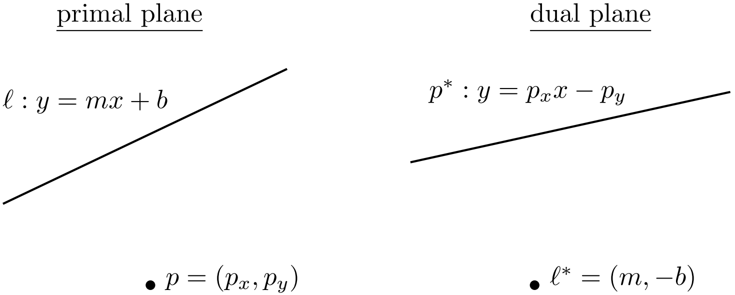


point $p = (p_x, p_y) \mapsto$ line $p^* : y = p_x x - p_y$

line $\ell : y = mx + b \mapsto$ point $\ell^* = (mx, -b)$

Note:

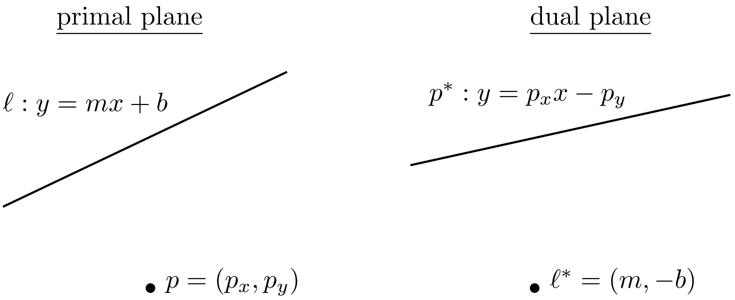
Duality



$$\text{point } p = (p_x, p_y) \mapsto \text{line } p^* : y = p_x x - p_y$$
$$\text{line } \ell : y = mx + b \mapsto \text{point } \ell^* = (mx, -b)$$

Note: self inverse $(p^*)^* = p, \quad (\ell^*)^* = \ell$

Duality

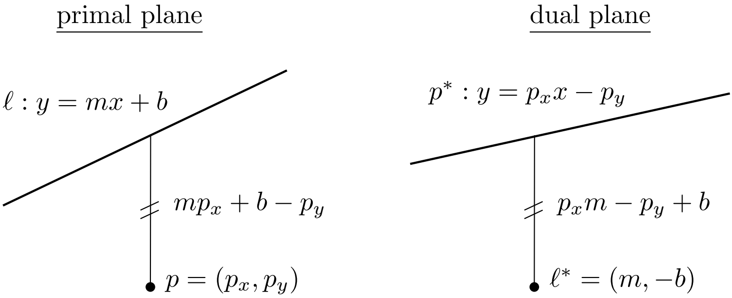


point $p = (p_x, p_y) \mapsto$ line $p^* : y = p_x x - p_y$

line $\ell : y = mx + b \mapsto$ point $\ell^* = (mx, -b)$

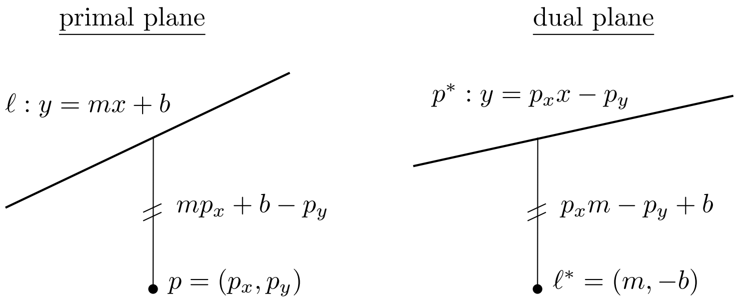
Note: does not handle vertical lines

Duality



duality preserves vertical distances

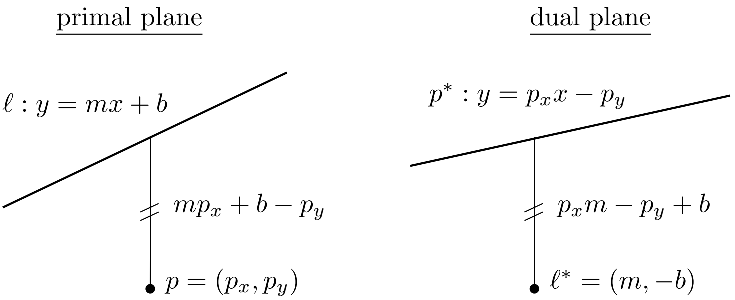
Duality



duality preserves vertical distances

$\Rightarrow$ incidence preserving: $p \in \ell$ if and only if $\ell^* \in p^*$

Duality



duality preserves vertical distances

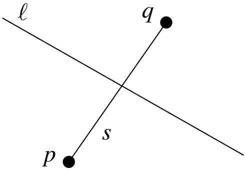
$\Rightarrow$ incidence preserving: $p \in \ell$ if and only if $\ell^* \in p^*$

$\Rightarrow$ order preserving: p lies above ℓ if and only if ℓ^* lies above p^*

Duality

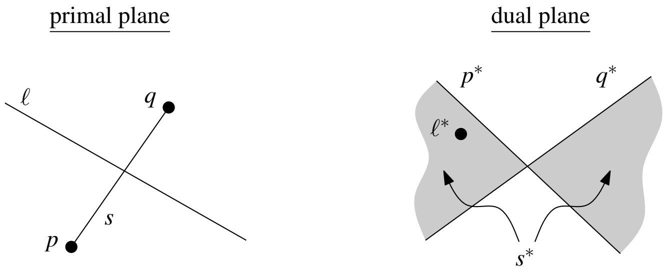
can be applied to other objects, e.g. segments

primal plane



Duality

can be applied to other objects, e.g. segments

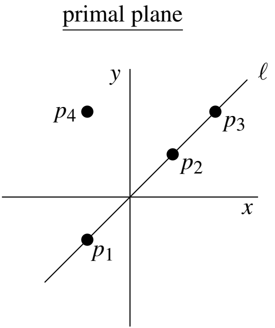


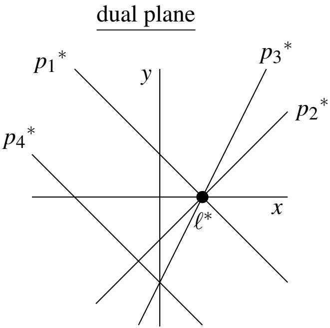
dual of a segment is a double wedge

Why use Duality?

It gives a new perspective!

E.g. 3 points on a line dualize to 3 lines intersecting in a point



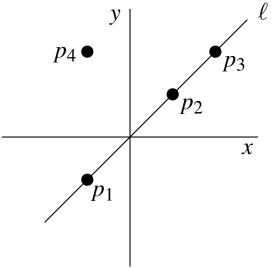


Why use Duality?

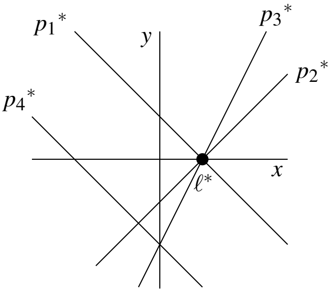
It gives a new perspective!

E.g. 3 points on a line dualize to 3 lines intersecting in a point

primal plane



dual plane

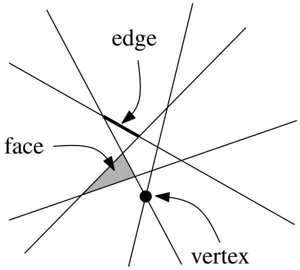


next we use arrangements

Arrangements of Lines

Arrangement $\mathcal{A}(L)$: subdivision induced by a set of lines L.

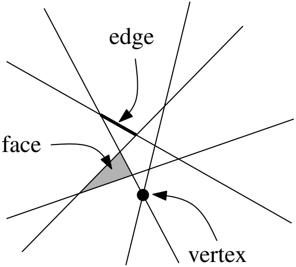
- consists of *faces, edges* and *vertices* (some unbounded)
- also arrangements of other geometric objects, e.g., segments, circles, higher-dimensional objects



Arrangements of Lines

Arrangement $\mathcal{A}(L)$: subdivision induced by a set of lines L.

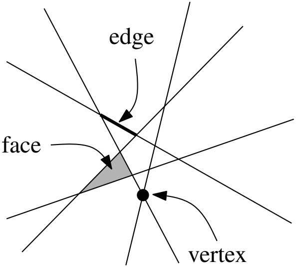
- consists of *faces, edges* and *vertices* (some unbounded)
- also arrangements of other geometric objects, e.g., segments, circles, higher-dimensional objects



Arrangements of Lines

Combinatorial Complexity:

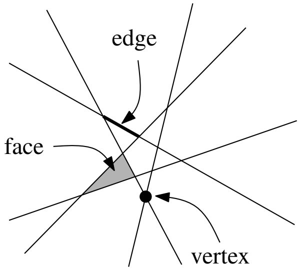
- $\leq n(n-1)/2$ vertices
- $\leq n^2$ edges
- $\leq n^2/2 + n/2 + 1$ faces: add lines incrementally $1 + \sum_{i=1}^{n} i = n(n+1)/2 + 1$
- equality holds in simple arrangements



Arrangements of Lines

Combinatorial Complexity:

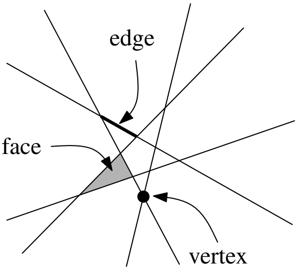
- $\leq n(n-1)/2$ vertices
- $\leq n^2$ edges
- $\leq n^2/2 + n/2 + 1$ faces:
  add lines incrementally
  $1 + \sum_{i=1}^{n} i = n(n+1)/2 + 1$
- equality holds in simple arrangements



Arrangements of Lines

Combinatorial Complexity:

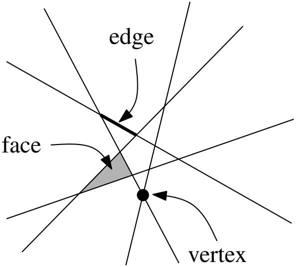
- $\leq n(n-1)/2$ vertices
- $\leq n^2$ edges
- $\leq n^2/2 + n/2 + 1$ faces: add lines incrementally $1 + \sum_{i=1}^{n} i = n(n+1)/2 + 1$
- equality holds in simple arrangements



Arrangements of Lines

Combinatorial Complexity:

- $\leq n(n-1)/2$ vertices
- $\leq n^2$ edges
- $\leq n^2/2 + n/2 + 1$ faces: add lines incrementally $1 + \sum_{i=1}^{n} i = n(n+1)/2 + 1$
- equality holds in simple arrangements

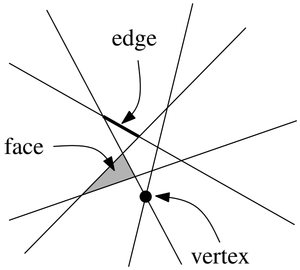


Arrangements of Lines

Combinatorial Complexity:

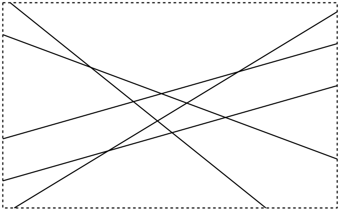
- $\leq n(n-1)/2$ vertices
- $\leq n^2$ edges
- $\leq n^2/2 + n/2 + 1$ faces: add lines incrementally $1 + \sum_{i=1}^{n} i = n(n+1)/2 + 1$
- equality holds in simple arrangements

overall $O(n^2)$ complexity



Constructing Arrangements

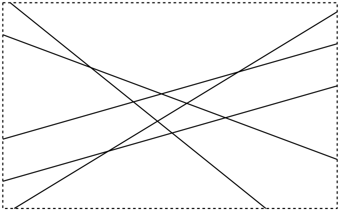
Goal: Compute $\mathcal{A}(L)$ in bounding box in DCEL representation



- plane sweep for line segment intersection: $O((n+k)\log n) = O(n^2 \log n)$
- faster: incremental construction

Constructing Arrangements

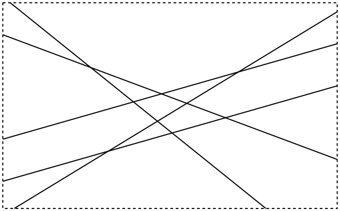
Goal: Compute $\mathcal{A}(L)$ in bounding box in DCEL representation



- plane sweep for line segment intersection: $O((n+k)\log n) = O(n^2 \log n)$
- faster: incremental construction

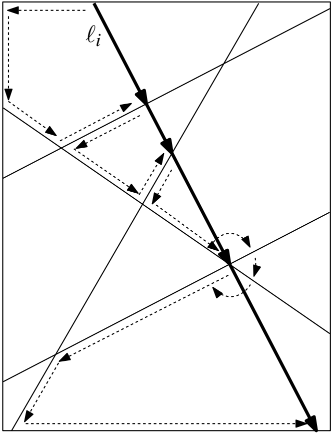
Constructing Arrangements

Goal: Compute $\mathcal{A}(L)$ in bounding box in DCEL representation



- plane sweep for line segment intersection:
  $O((n+k)\log n) = O(n^2 \log n)$
- faster: incremental construction

Incremental Construction



Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

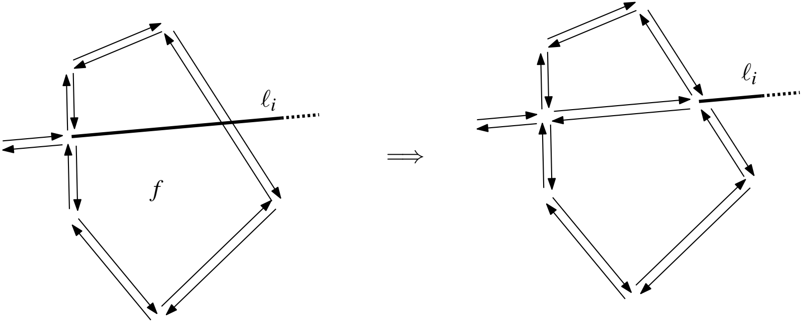
Incremental Construction

Algorithm CONSTRUCTARRANGEMENT(L)
Input. A set L of n lines in the plane.
Output. DCEL for subdivision induced by $\mathcal{B}(L)$ and the part of $\mathcal{A}(L)$ inside $\mathcal{B}(L)$, where $\mathcal{B}(L)$ is a suitable bounding box.

1. Compute a bounding box $\mathcal{B}(L)$ that contains all vertices of $\mathcal{A}(L)$ in its interior.
2. Construct DCEL for the subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4.   **do** Find the edge e on $\mathcal{B}(L)$ that contains the leftmost intersection point of ℓ_i and $\mathcal{A}_i$.
5. $f \leftarrow$ the bounded face incident to e
6.   **while** f is not the unbounded face, that is, the face outside $\mathcal{B}(L)$
7.   **do** Split f, and set f to be the next intersected face.

Incremental Construction

Face split:



Incremental Construction

Runtime analysis:

Algorithm ConstructArrangement(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

Runtime analysis:

1. $O(n^2)$

Algorithm ConstructArrangement(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

Runtime analysis:

1. $O(n^2)$
2. constant

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

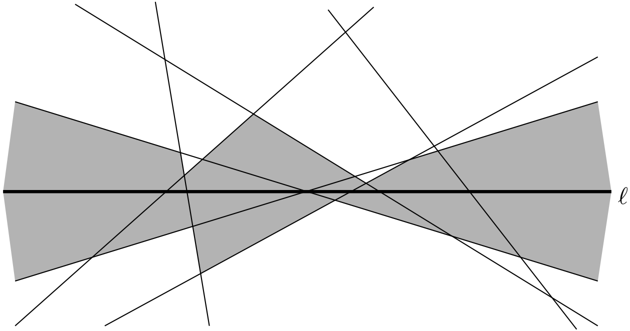
Runtime analysis:

1. $O(n^2)$
2. constant
3. ?

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

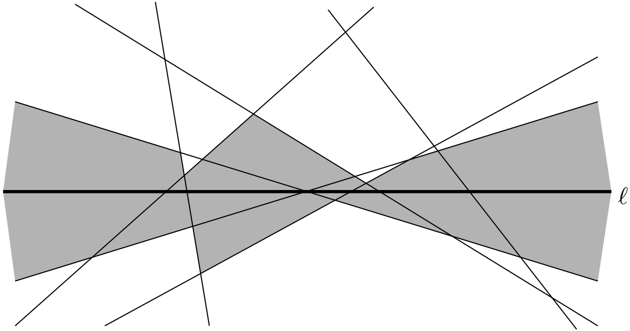
Zone Theorem

The zone of a line ℓ in an arrangement $\mathcal{A}(L)$ is the set of faces of $\mathcal{A}(L)$ whose closure intersects ℓ.



Zone Theorem

The zone of a line ℓ in an arrangement $\mathcal{A}(L)$ is the set of faces of $\mathcal{A}(L)$ whose closure intersects ℓ.



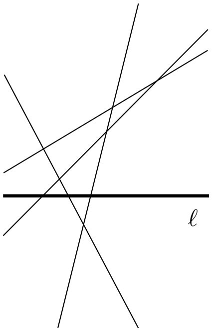
Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Zone Theorem

Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Proof:

- We can assume ℓ horizontal and no other line horizontal.
- We count number of *left-bounding* edges.
- We show by induction on m that this at most $5m$:
  - $m = 1$: trivially true
  - $m > 1$: only at most 3 new edges if ℓ_1 is unique;

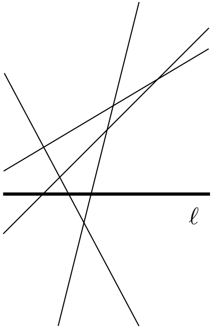


Zone Theorem

Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Proof:

- We can assume ℓ horizontal and no other line horizontal.
- We count number of *left-bounding* edges.
- We show by induction on m that this at most $5m$:
  - $m = 1$: trivially true
  - $m > 1$: only at most 3 new edges if ℓ_1 is unique,

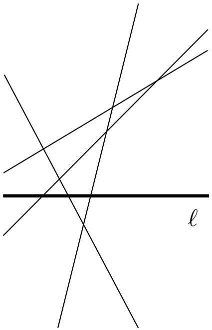


Zone Theorem

Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Proof:

- We can assume ℓ horizontal and no other line horizontal.
- We count number of *left-bounding* edges.
- We show by induction on m that this at most $5m$:
  - $m = 1$: trivially true
  - $m > 1$: only at most 3 new edges if ℓ_1 is unique,

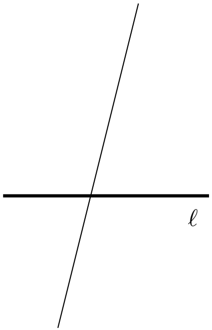


Zone Theorem

Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Proof:

- We can assume ℓ horizontal and no other line horizontal.
- We count number of *left-bounding* edges.
- We show by induction on m that this at most $5m$:
  - $m = 1$: trivially true
  - $m > 1$: only at most 3 new edges if ℓ_1 is unique,

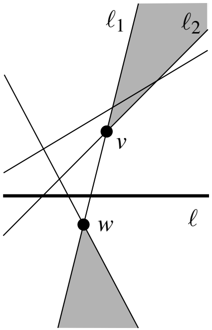


Zone Theorem

Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Proof:

- We can assume ℓ horizontal and no other line horizontal.
- We count number of *left-bounding* edges.
- We show by induction on m that this at most $5m$:
  - $m = 1$: trivially true
  - $m > 1$: only at most 3 new edges if ℓ_1 is unique,

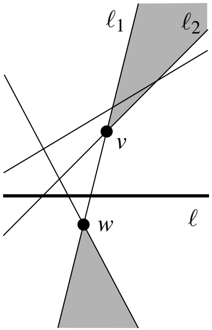


Zone Theorem

Theorem: The complexity of the zone of a line in an arrangement of m lines is $O(m)$.

Proof:

- We can assume ℓ horizontal and no other line horizontal.
- We count number of *left-bounding* edges.
- We show by induction on m that this at most $5m$:
  - $m = 1$: trivially true
  - $m > 1$: only at most 3 new edges if ℓ_1 is unique, at most 5 if ℓ_1 is not unique. $5(m-1)+5 = 5m$



Incremental Construction

.

Run time analysis:

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

.

Run time analysis:

1. $O(n^2)$

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

.

Run time analysis:

1. $O(n^2)$
2. constant

Algorithm ConstructArrangement(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

.

Run time analysis:

1. $O(n^2)$
2. constant
3. $\sum_{i=1}^{n} O(i) = O(n^2)$

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

Incremental Construction

.

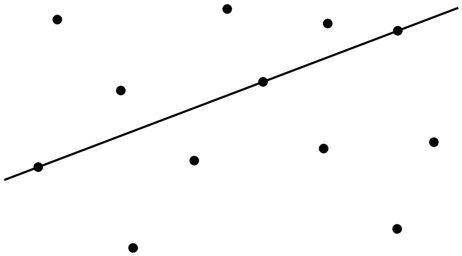
Run time analysis:

1. $O(n^2)$
2. constant
3. $\sum_{i=1}^{n} O(i) = O(n^2)$

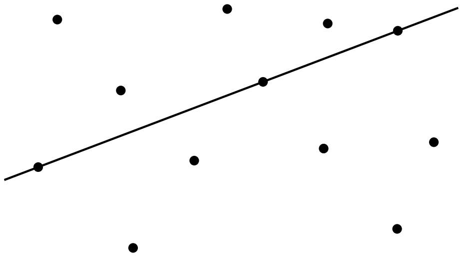
in total $O(n^2)$

Algorithm CONSTRUCTARRANGEMENT(L)
Input. Set L of n lines.
Output. DCEL for $\mathcal{A}(L)$ in $\mathcal{B}(L)$.
1. Compute bounding box $\mathcal{B}(L)$.
2. Construct DCEL for subdivision induced by $\mathcal{B}(L)$.
3. **for** $i \leftarrow 1$ **to** n
4. **do** insert ℓ_i.

3 Points on a Line



3 Points on a Line



Algorithm:

- run incremental construction algorithm for dual problem
- stop when 3 lines pass through a point

Run time: $O(n^2)$