Counting Convex k-gons in an Arrangement of Line Segments

Martin Fink, Neeraj Kumar and Subhash Suri

University of California, Santa Barbara

Motivation

Consider the following problem from computer vision:

Motivation

Consider the following problem from computer vision:

Given a camera image I representing object boundaries, estimate the number of rectangular objects in the scene.

Motivation

Consider the following problem from computer vision:

Given a camera image I representing object boundaries, estimate the number of rectangular objects in the scene.

- Camera image $I \Rightarrow$ Arrangement $\mathcal{A}$ of line segments

Motivation

Consider the following problem from computer vision:

Given a camera image I representing object boundaries, estimate the number of rectangular objects in the scene.

- Camera image I $\Rightarrow$ Arrangement $\mathcal{A}$ of line segments
- Perspective transformation:
  Rectangles in scene $\Rightarrow$ quadilaterals in image

Motivation

Consider the following problem from computer vision:

Given a camera image I representing object boundaries, estimate the number of rectangular objects in the scene.

- Camera image $I \Rightarrow$ Arrangement $\mathcal{A}$ of line segments
- Perspective transformation:
  Rectangles in scene $\Rightarrow$ quadilaterals in image
- Count all convex quadilaterals (4-gon) in an arrangement

Motivation

Consider the following problem from computer vision:

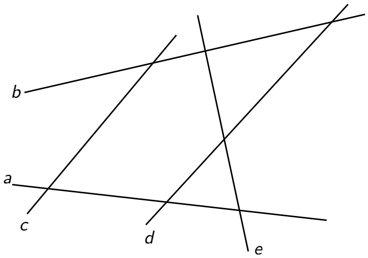
Given a camera image I representing object boundaries, estimate the number of rectangular objects in the scene.

- Camera image $I \Rightarrow$ Arrangement $\mathcal{A}$ of line segments
- Perspective transformation:
  Rectangles in scene $\Rightarrow$ quadilaterals in image
- Count all convex quadilaterals (4-gon) in an arrangement

Natural generalization to convex k-gons

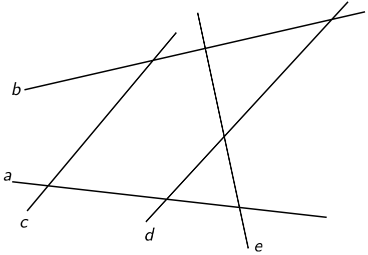
Problem Definition

Given: An arrangement $\mathcal{A}(S)$ of line segments S in 2-D



Problem Definition

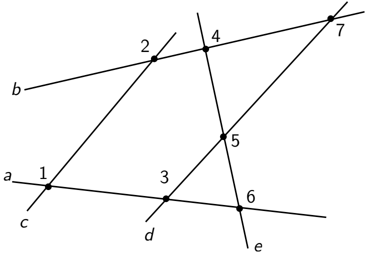
Given: An arrangement $\mathcal{A}(S)$ of line segments S in 2-D



A convex k-gon of $\mathcal{A}(S)$ is a convex polygon with k sides if:

Problem Definition

Given: An arrangement $\mathcal{A}(S)$ of line segments S in 2-D

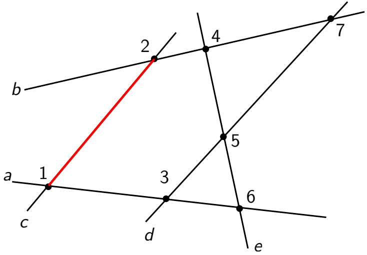


A convex k-gon of $\mathcal{A}(S)$ is a convex polygon with k sides if:

- vertices are a subset of arrangement vertices.

Problem Definition

Given: An arrangement $\mathcal{A}(S)$ of line segments S in 2-D

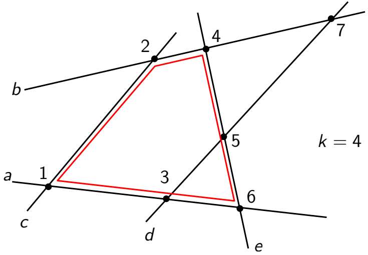


A convex k-gon of $\mathcal{A}(S)$ is a convex polygon with k sides if:

- vertices are a subset of arrangement vertices.
- sides are part of input segments.

Problem Definition

Given: An arrangement $\mathcal{A}(S)$ of line segments S in 2-D

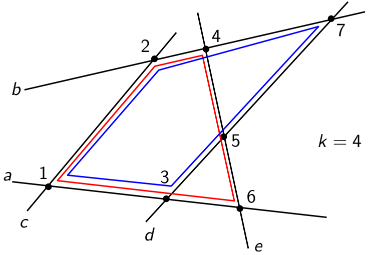


A convex k-gon of $\mathcal{A}(S)$ is a convex polygon with k sides if:

- vertices are a subset of arrangement vertices.
- sides are part of input segments.

Problem Definition

Given: An arrangement $\mathcal{A}(S)$ of line segments S in 2-D



A convex k-gon of $\mathcal{A}(S)$ is a convex polygon with k sides if:

- vertices are a subset of arrangement vertices.
- sides are part of input segments.

Goal: count and report all such k-gons.

Our Results

- Count all k-gons in $O(n \log n + mn)$ time and $O(n^2)$ space (for constant k)

Our Results

- Count all k-gons in $O(n \log n + mn)$ time and $O(n^2)$ space (for constant k)
- Report set of all k-gons K in $O(|K|)$ additional time and $O(mn)$ additional space

Our Results

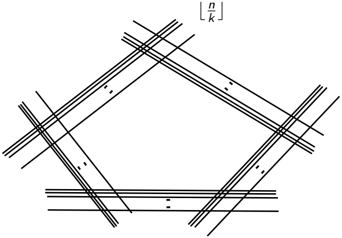
- Count all k-gons in $O(n \log n + mn)$ time and $O(n^2)$ space (for constant k)
- Report set of all k-gons K in $O(|K|)$ additional time and $O(mn)$ additional space

Count in time much faster than the number of k-gons :

Our Results

- Count all k-gons in $O(n \log n + mn)$ time and $O(n^2)$ space (for constant k)
- Report set of all k-gons K in $O(|K|)$ additional time and $O(mn)$ additional space

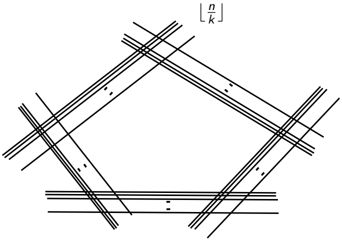
Count in time much faster than the number of k-gons : $\Theta(n^k)$



Our Results

- Count all k-gons in $O(n \log n + mn)$ time and $O(n^2)$ space (for constant k)
- Report set of all k-gons K in $O(|K|)$ additional time and $O(mn)$ additional space

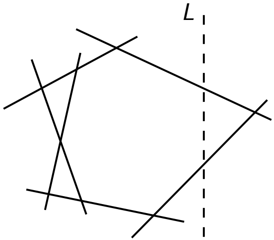
Count in time much faster than the number of k-gons : $\Theta(n^k)$



- Counting k-gons is as hard as the 3SUM problem, for $k = 3, 4$

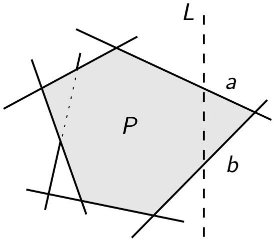
Counting k-gons

A vertical line L intersects at most two sides of a k-gon P.



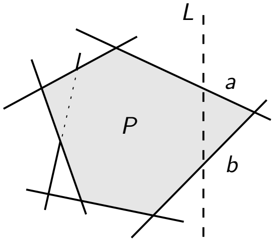
Counting k-gons

A vertical line L intersects at most two sides of a k-gon P.



Counting k-gons

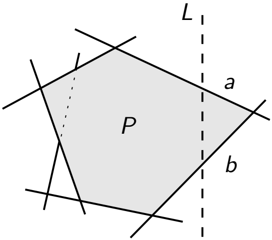
A vertical line L intersects at most two sides of a k-gon P.



$\mathrm{span}(P, L) = (a, b)$

Counting k-gons

A vertical line L intersects at most two sides of a k-gon P.

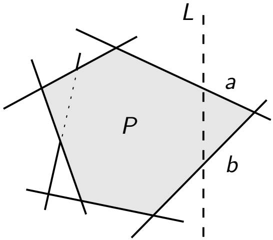


$\mathrm{span}(P, L) = (a, b)$

- $O(n^2)$ distinct spans (w.r.t. L) among all k-gons

Counting k-gons

A vertical line L intersects at most two sides of a k-gon P.

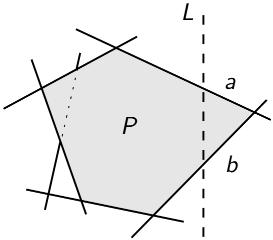


$\text{span}(P, L) = (a, b)$

- $O(n^2)$ distinct spans (w.r.t. L) among all k-gons
- Suggests a plane sweep based algorithm, key idea:

Counting k-gons

A vertical line L intersects at most two sides of a k-gon P.

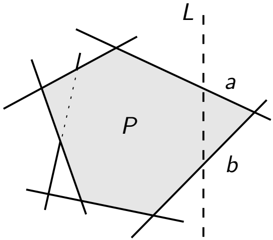


$\mathrm{span}(P, L) = (a, b)$

- $O(n^2)$ distinct spans (w.r.t. L) among all k-gons
- Suggests a plane sweep based algorithm, key idea:
  - Assign a k-gon intersecting L to its span

Counting k-gons

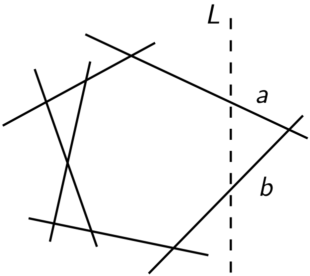
A vertical line L intersects at most two sides of a k-gon P.



$\text{span}(P, L) = (a, b)$

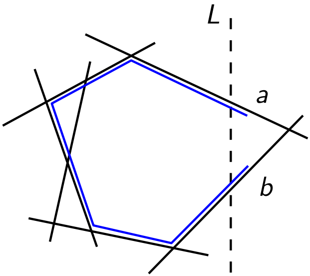
- $O(n^2)$ distinct spans (w.r.t. L) among all k-gons
- Suggests a plane sweep based algorithm, key idea:
  - Assign a k-gon intersecting L to its span
  - Update count as we sweep L across the plane

Notation



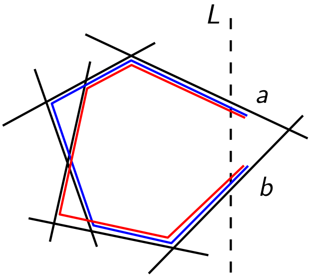
- **Open *j*-gons**: All $j \leq k$ sides start left of L

Notation



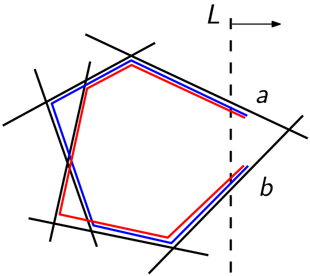
- **Open j-gons**: All $j \leq k$ sides start left of L

Notation



- **Open j-gons**: All $j \leq k$ sides start left of L
  - $\sigma(a, b, j)$: Number of open j-gons with span (a, b)

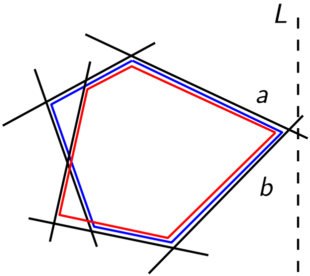
Notation



$\sigma(a, b, 5) = 2$

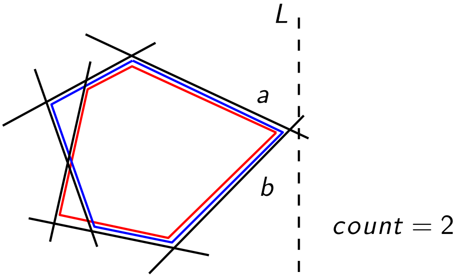
- **Open j-gons**: All $j \leq k$ sides start left of L
  - $\sigma(a, b, j)$: Number of open j-gons with span (a, b)

Notation



- **Open j-gons**: All $j \leq k$ sides start left of L
  - $\sigma(a, b, j)$: Number of open j-gons with span (a, b)
- **Closed k-gons**: All k sides end left of L

Notation



- **Open j-gons**: All $j \leq k$ sides start left of L
  - $\sigma(a, b, j)$: Number of open j-gons with span (a, b)
- **Closed k-gons**: All k sides end left of L
  - *count*: number of k-gons left of L

Algorithm Steps

- Set $count = 0$ and $\sigma(a, b, j) = 0$, for all a, b, j

Algorithm Steps

- Set $count = 0$ and $\sigma(a, b, j) = 0$, for all a, b, j
- Compute all intersections (Event points)

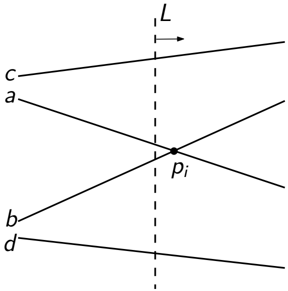
Algorithm Steps

- Set $count = 0$ and $\sigma(a, b, j) = 0$, for all a, b, j
- Compute all intersections (Event points)
- For each event from left to right: Perform Updates

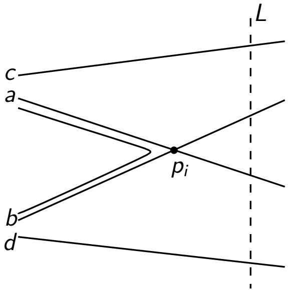
Algorithm Steps

- Set *count* $= 0$ and $\sigma(a, b, j) = 0$, for all a, b, j
- Compute all intersections (Event points)
- For each event from left to right: Perform Updates
- Return *count*

Updates at intersection (a, b)

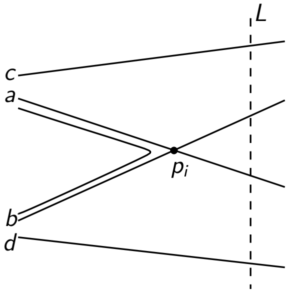


Updates at intersection (a, b)



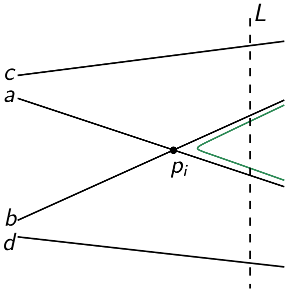
- Some k-gons complete

Updates at intersection (a, b)



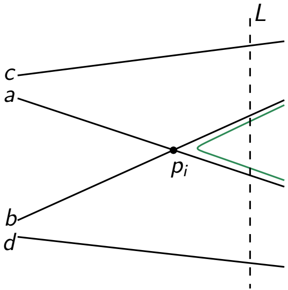
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$

Updates at intersection (a, b)



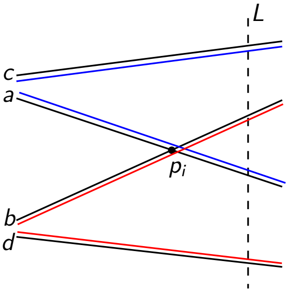
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins

Updates at intersection (a, b)



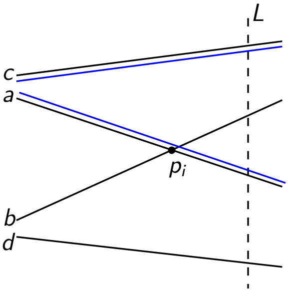
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) = 1$

Updates at intersection (a, b)



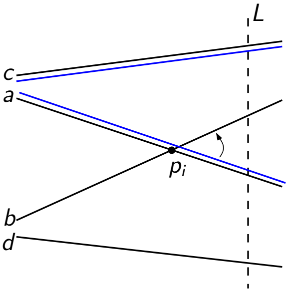
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) \;=\; 1$
- Some j-gons grow into $j+1$-gons

Updates at intersection (a, b)



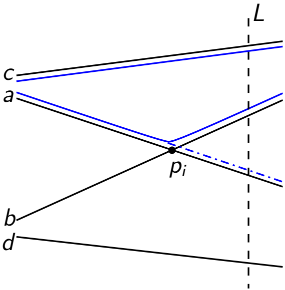
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) \; = \; 1$
- Some j-gons grow into $j + 1$-gons
  - For all segments c *above* a

Updates at intersection (a, b)



- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) \; = \; 1$
- Some j-gons grow into $j+1$-gons
  - For all segments c *above* a

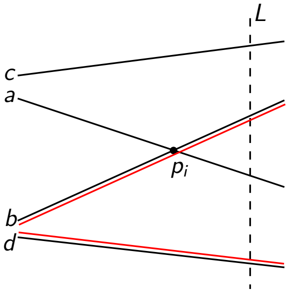
Updates at intersection (a, b)



- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) \;=\; 1$
- Some j-gons grow into $j+1$-gons
  - For all segments c *above* a

$$\sigma(c, b, j+1) \mathrel{+}= \sigma(c, a, j)$$

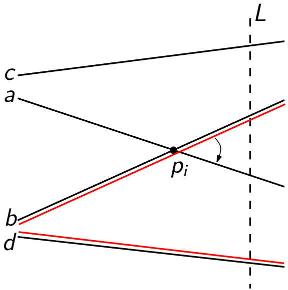
Updates at intersection (a, b)



- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) \; = \; 1$
- Some j-gons grow into $j+1$-gons
  - For all segments c *above* a

    $$\sigma(c, b, j+1) \mathrel{+}= \sigma(c, a, j)$$
  - For all segments d *below* b

Updates at intersection (a, b)

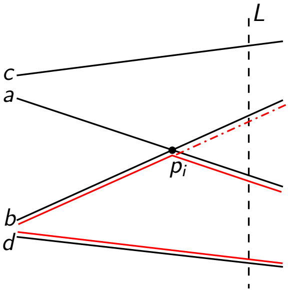


- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) \;=\; 1$
- Some j-gons grow into $j+1$-gons
  - For all segments c *above* a

    $$\sigma(c, b, j+1) \mathrel{+}= \sigma(c, a, j)$$

  - For all segments d *below* b

Updates at intersection (a, b)



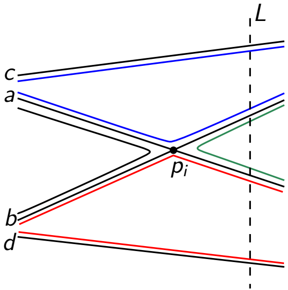
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) = 1$
- Some j-gons grow into $j + 1$-gons
  - For all segments c *above* a

    $$\sigma(c, b, j + 1) \mathrel{+}= \sigma(c, a, j)$$

  - For all segments d *below* b

    $$\sigma(a, d, j + 1) \mathrel{+}= \sigma(b, d, j)$$

Updates at intersection (a, b)



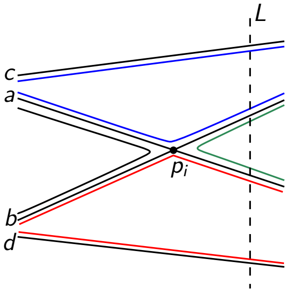
- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) = 1$
- Some j-gons grow into $j+1$-gons
  - For all segments c *above* a

    $$\sigma(c, b, j+1) \mathrel{+}= \sigma(c, a, j)$$
  - For all segments d *below* b

    $$\sigma(a, d, j+1) \mathrel{+}= \sigma(b, d, j)$$

Total $O(n)$ time per intersection

Updates at intersection (a, b)



- Some k-gons complete
  - $count \mathrel{+}= \sigma(a, b, k)$
- A 2-gon begins
  - $\sigma(b, a, 2) = 1$
- Some j-gons grow into $j + 1$-gons
  - For all segments c *above* a

    $$\sigma(c, b, j + 1) \mathrel{+}= \sigma(c, a, j)$$
  - For all segments d *below* b

    $$\sigma(a, d, j + 1) \mathrel{+}= \sigma(b, d, j)$$

Total $O(n)$ time per intersection

Handles degenerate cases: apply pairwise updates collectively

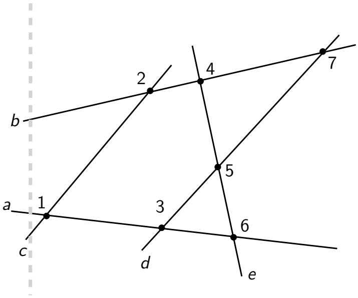
Reporting Problem

Goal: Report all k-gons of the output set K.

Reporting Problem

Goal: Report all k-gons of the output set K.

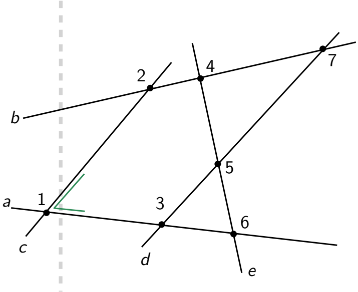
Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



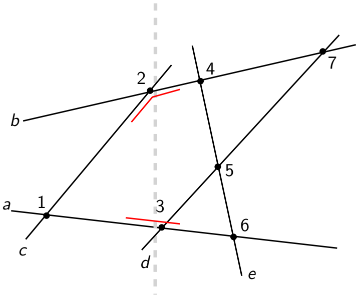
$(c, a, 2)$

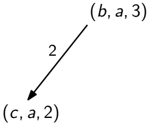
Create a vertex for the new open 2-gon

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



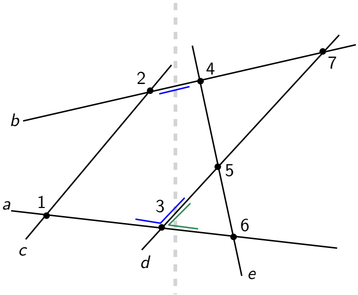


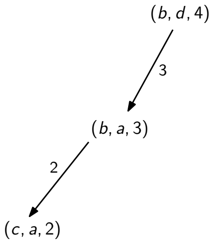
Add an edge for a j-gon growing into $j+1$-gon

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



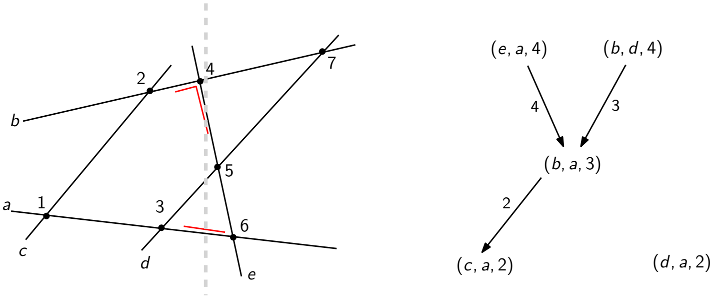


Add an edge for a j-gon growing into $j + 1$-gon

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$

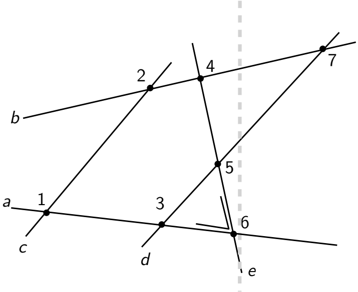


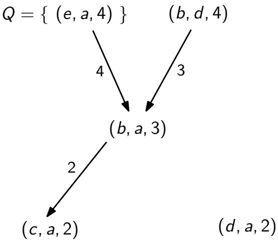
Add an edge for a j-gon growing into $j+1$-gon

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



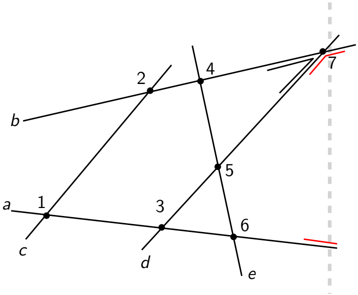


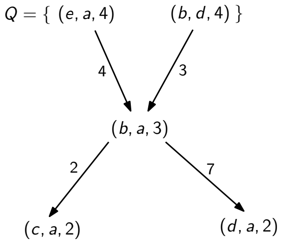
Append vertices for completed k-gons to Q

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



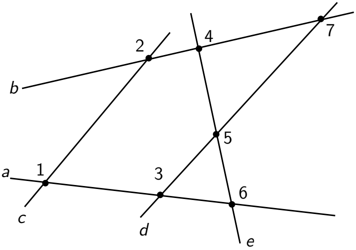


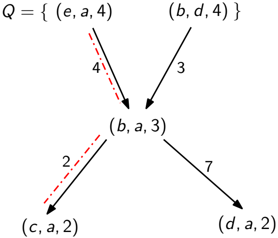
Append vertices for completed k-gons to Q

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



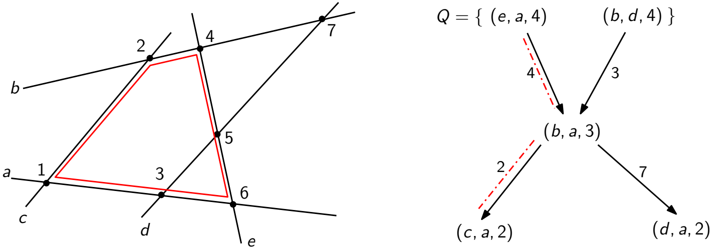


Recursively enumerate all k-gons

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$

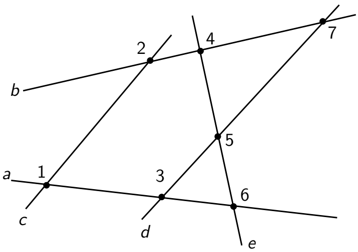


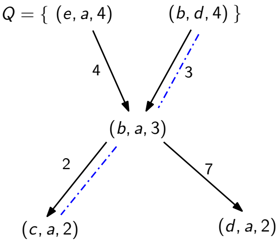
Recursively enumerate all k-gons

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



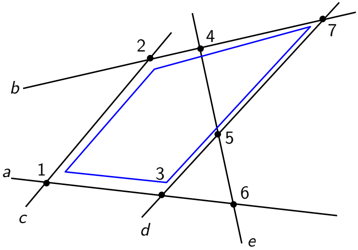


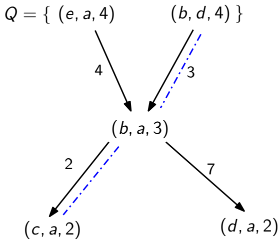
Recursively enumerate all k-gons

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



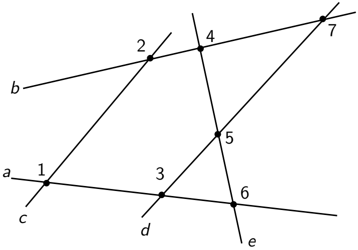


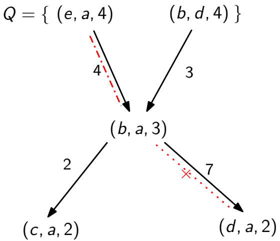
Recursively enumerate all k-gons

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



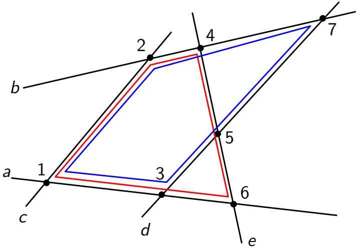


Only time respecting paths are valid k-gons

Reporting Problem

Goal: Report all k-gons of the output set K.

Keep track of updates using acyclic digraph $G = (V, E, \mathcal{L})$



$K = \{(1, 2, 4, 6), \quad (1, 2, 3, 7)\}$

Report all k-gons in $O(|K|)$ additional time

3SUM Hardness

Reduction from POINT-ON-3-LINES problem

3SUM Hardness

Reduction from POINT-ON-3-LINES problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

3SUM Hardness

Reduction from $\textsc{Point-on-3-lines}$ problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-}\textsc{Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

3SUM Hardness

Reduction from $\textsc{Point-on-3-lines}$ problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-}\textsc{Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel

3SUM Hardness

Reduction from $\textsc{Point-on-3-lines}$ problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-}\textsc{Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel
- Compute bounding box B of the arrangement

3SUM Hardness

Reduction from $\textsc{Point-on-3-lines}$ problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-}\textsc{Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel
- Compute bounding box B of the arrangement
- Clip the lines around B to obtain an arrangement of segments

3SUM Hardness

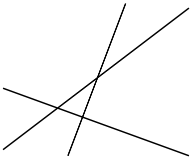
Reduction from $\textsc{Point-on-3-lines}$ problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-}\textsc{Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel
- Compute bounding box B of the arrangement
- Clip the lines around B to obtain an arrangement of segments

$\binom{n}{3}$ triangles $\Leftrightarrow$ no $\textsc{Point-on-3-lines}$



3SUM Hardness

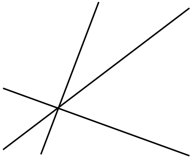
Reduction from Point-on-3-lines problem

Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-Points-on-line} \xrightarrow{\text{dual}} \text{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel
- Compute bounding box B of the arrangement
- Clip the lines around B to obtain an arrangement of segments

$\binom{n}{3}$ triangles $\Leftrightarrow$ no Point-on-3-lines



3SUM Hardness

Reduction from $\textsc{Point-on-3-lines}$ problem

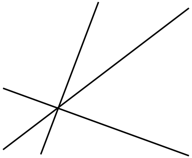
Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \text{3-}\textsc{Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel
- Compute bounding box B of the arrangement
- Clip the lines around B to obtain an arrangement of segments

$\binom{n}{3}$ triangles $\Leftrightarrow$ no $\textsc{Point-on-3-lines}$

$\binom{n}{4}$ quadilaterals $\Leftrightarrow$ no $\textsc{Point-on-3-lines}$



3SUM Hardness

Reduction from $\textsc{Point-on-3-lines}$ problem

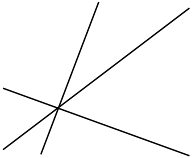
Given a set of lines L in plane, is there a point that lies on 3 lines?

$$\text{3SUM} \longrightarrow \textsc{3-Points-on-line} \xrightarrow{\text{dual}} \textsc{Point-on-3-lines}$$

- Reduction ensures that no two lines in L are parallel
- Compute bounding box B of the arrangement
- Clip the lines around B to obtain an arrangement of segments

$\binom{n}{3}$ triangles $\Leftrightarrow$ no $\textsc{Point-on-3-lines}$

$\binom{n}{4}$ quadilaterals $\Leftrightarrow$ no $\textsc{Point-on-3-lines}$



Does not extend to $k \geq 5$

Concluding Remarks

Concluding Remarks

- Introduced the k-gon counting problem

Concluding Remarks

- Introduced the k-gon counting problem
- Algorithm for k-gon counting in $O(mn) \in O(n^3)$ time

Concluding Remarks

- Introduced the k-gon counting problem
- Algorithm for k-gon counting in $O(mn) \in O(n^3)$ time
- Reporting in additional $O(|K|)$ time

Concluding Remarks

- Introduced the k-gon counting problem
- Algorithm for k-gon counting in $O(mn) \in O(n^3)$ time
- Reporting in additional $O(|K|)$ time
- 3SUM hardness for $k = 3, 4 \Rightarrow$ Significantly better than $O(n^2)$ unlikely

Concluding Remarks

- Introduced the k-gon counting problem
- Algorithm for k-gon counting in $O(mn) \in O(n^3)$ time
- Reporting in additional $O(|K|)$ time
- 3SUM hardness for $k = 3, 4 \Rightarrow$ Significantly better than $O(n^2)$ unlikely
- Open question: faster algorithms?

Thanks!