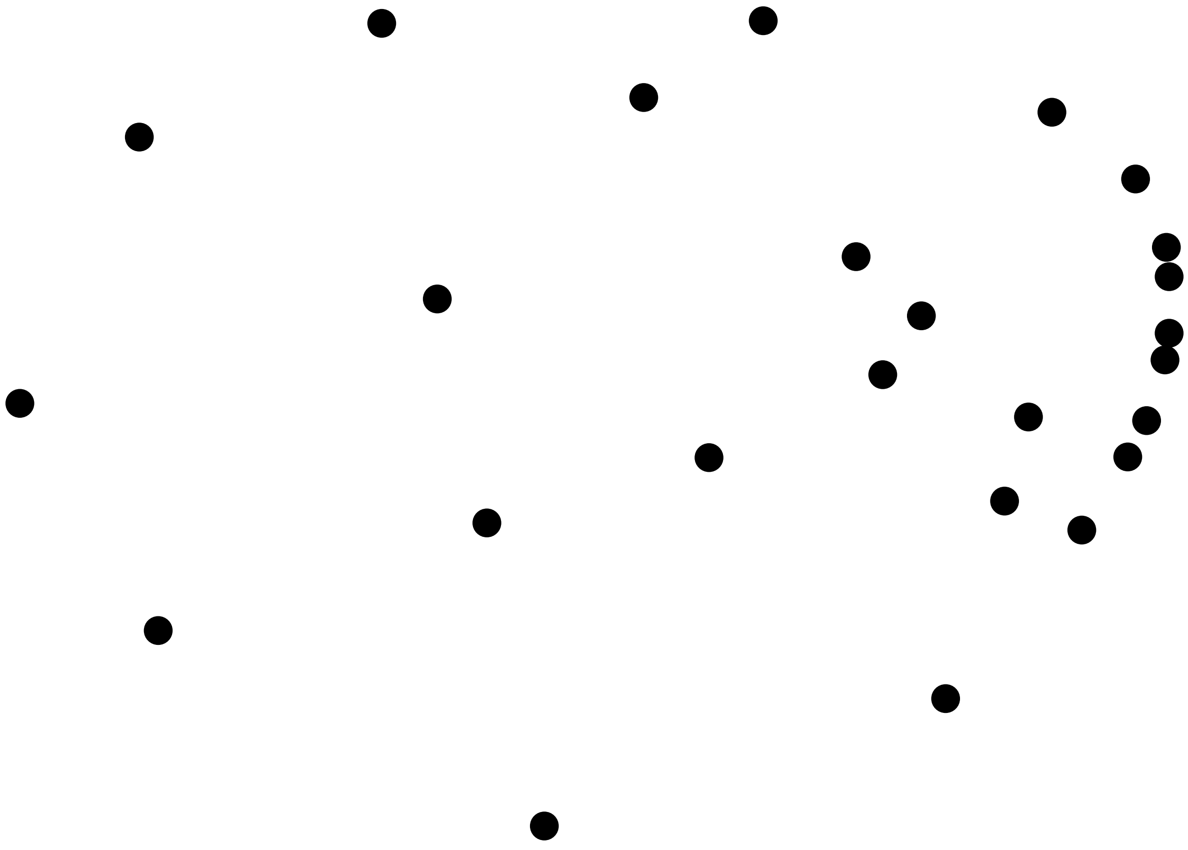
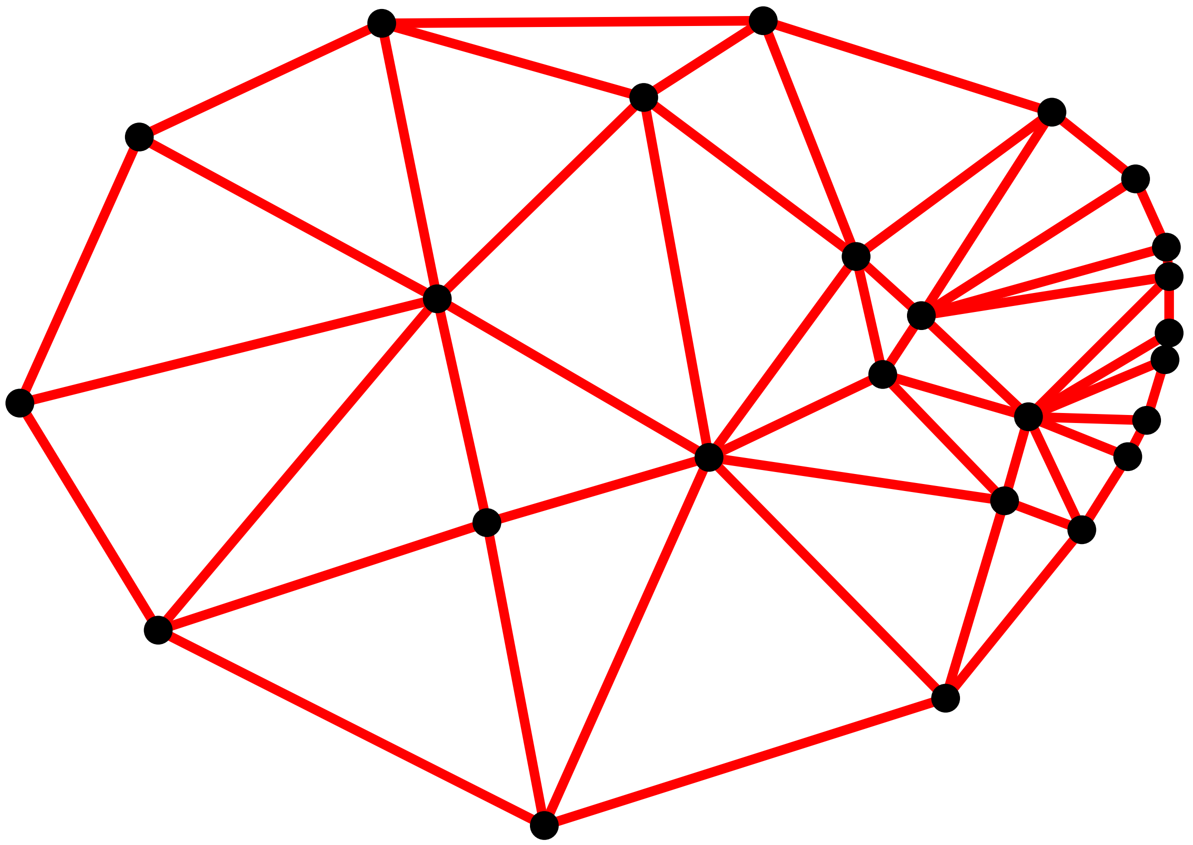
Delaunay Triangulation

Delaunay Triangulation



Delaunay Triangulation



Delaunay Triangulation

- Pencils of circles
- Definition, empty circle property
- Motivations: reconstruction, meshing
- Properties: EMST, max-min angle
- Lower bound
- In-sphere predicate
- Diagonal flipping
- Incremental algorithm
- Sweep line algorithm
- Miscellaneous

Delaunay Triangulation: pencils of circles

Imagine moving circles

Delaunay Triangulation: pencils of circles

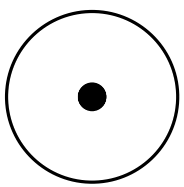
Imagine moving circles

fixed center

increasing radius

Delaunay Triangulation: pencils of circles

Imagine moving circles

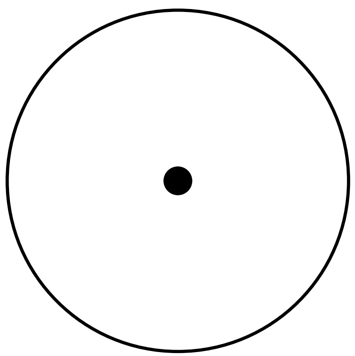


fixed center

increasing radius

Delaunay Triangulation: pencils of circles

Imagine moving circles

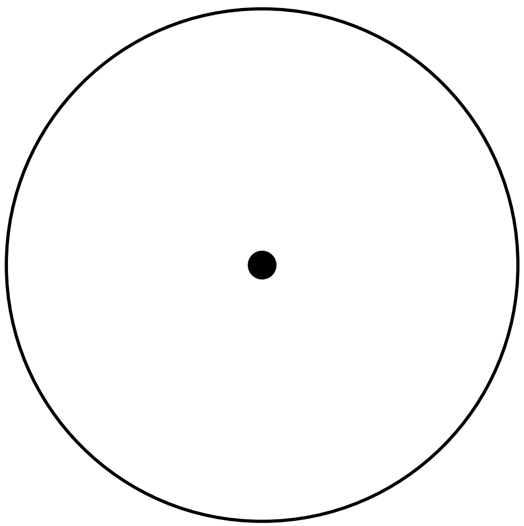


fixed center

increasing radius

Delaunay Triangulation: pencils of circles

Imagine moving circles

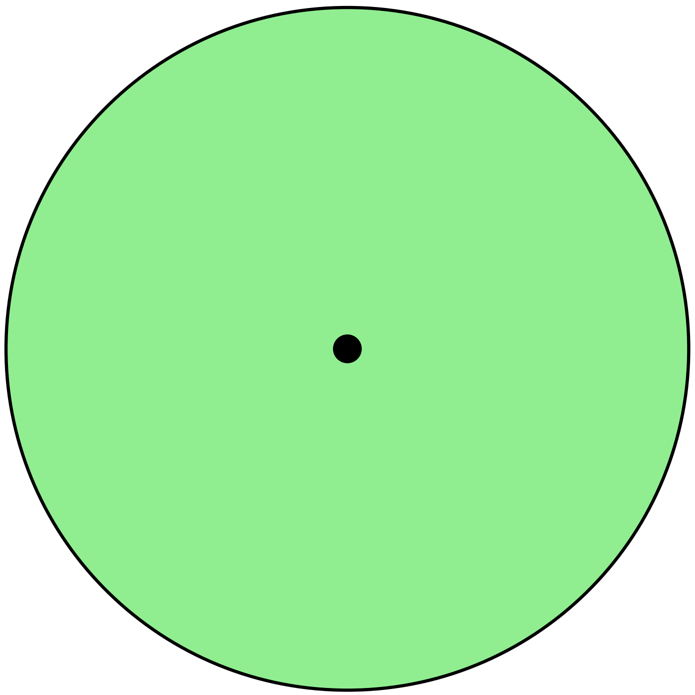


fixed center

increasing radius

Delaunay Triangulation: pencils of circles

Imagine moving circles



fixed center

increasing radius

Cocentric pencil

Delaunay Triangulation: pencils of circles

Imagine moving circles

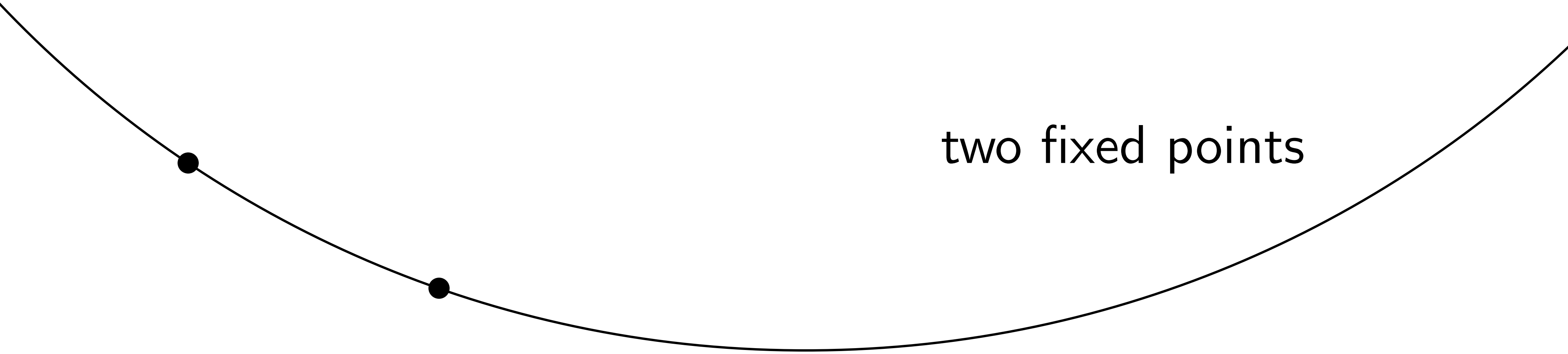
Delaunay Triangulation: pencils of circles

Imagine moving circles

two fixed points

Delaunay Triangulation: pencils of circles

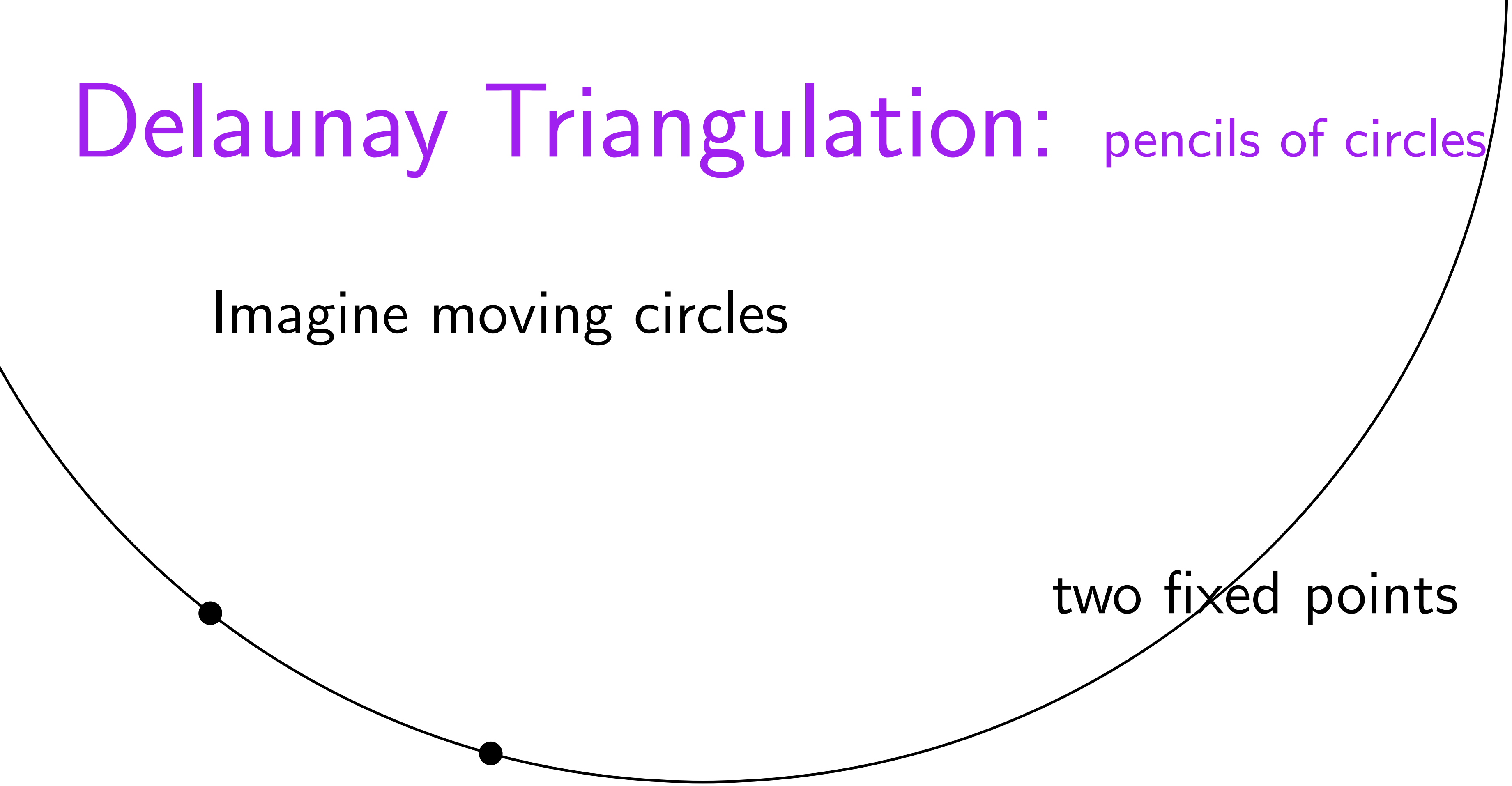
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

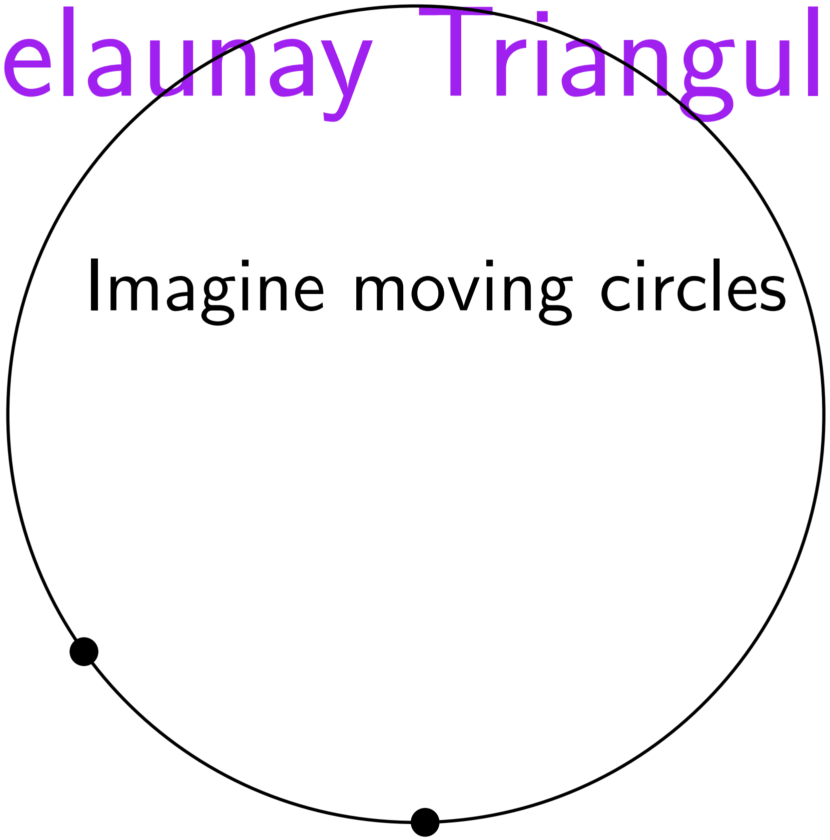
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

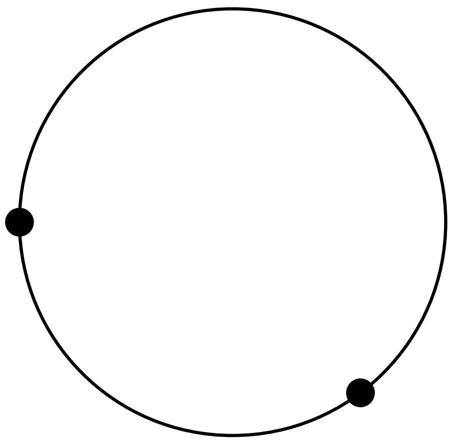
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

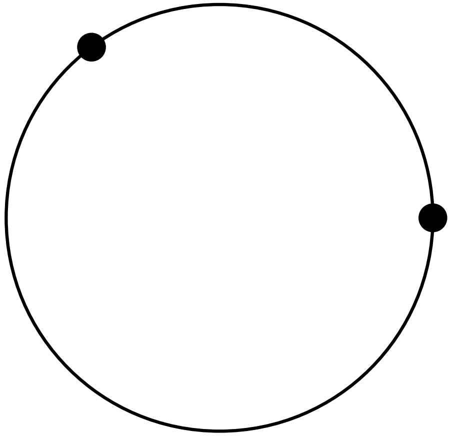
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

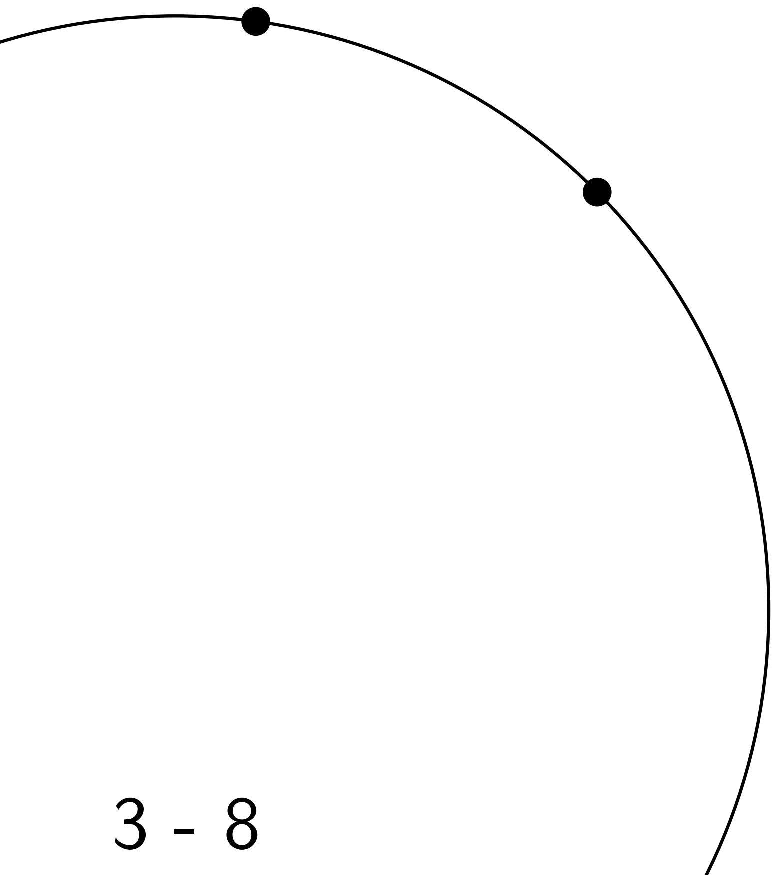
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

Imagine moving circles

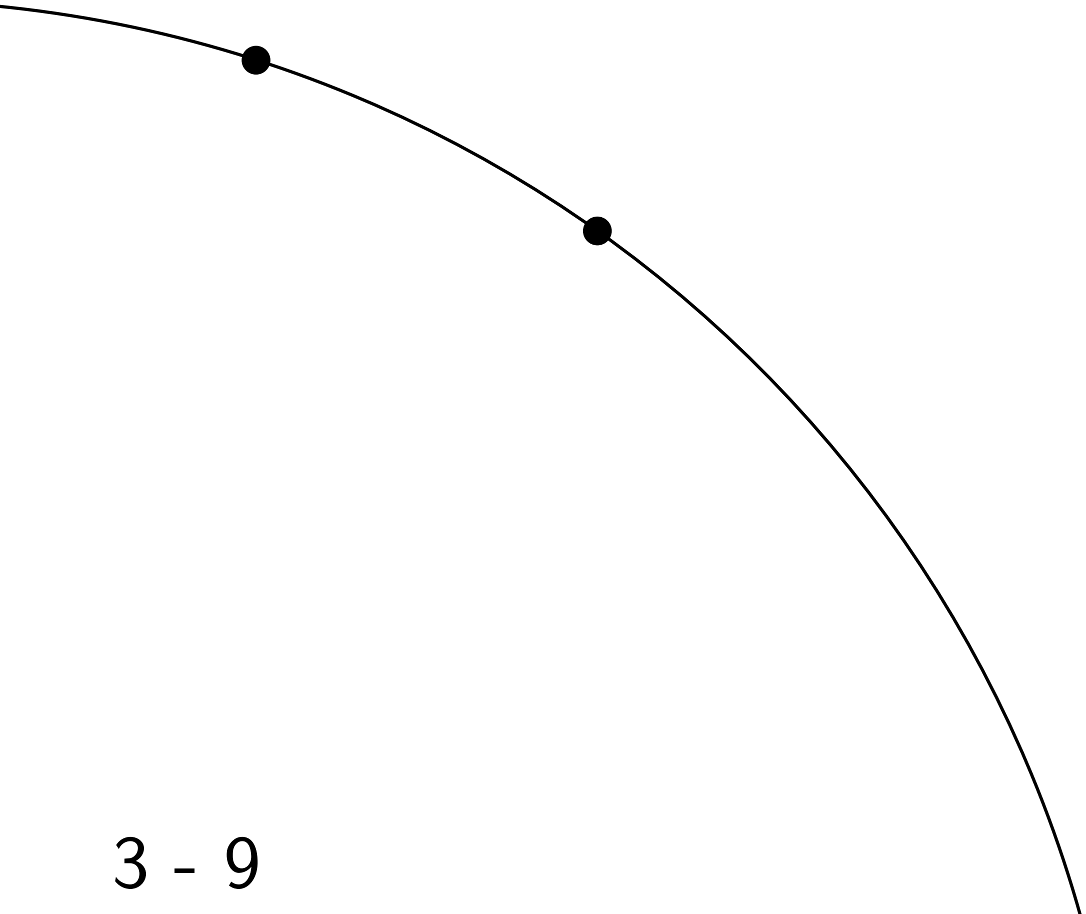


two fixed points

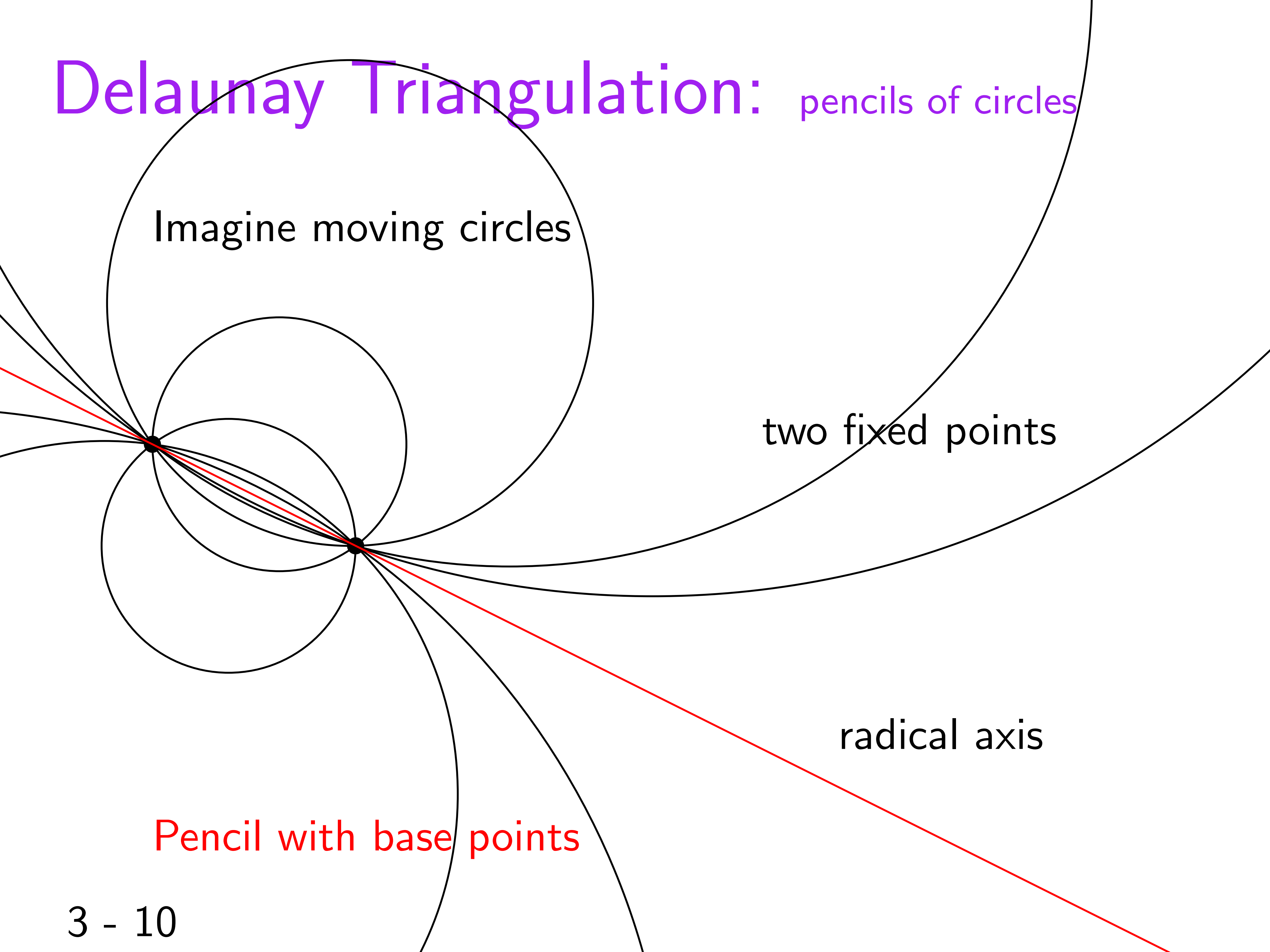
Delaunay Triangulation: pencils of circles

Imagine moving circles

two fixed points



Delaunay Triangulation: pencils of circles



Delaunay Triangulation: pencils of circles

Imagine moving circles

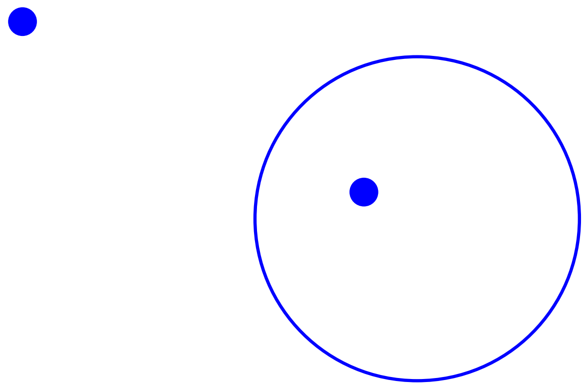
Delaunay Triangulation: pencils of circles

Imagine moving circles

two fixed points

Delaunay Triangulation: pencils of circles

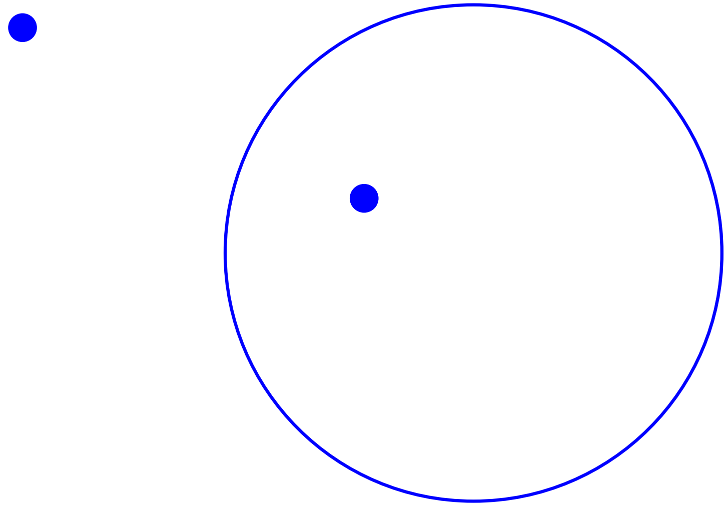
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

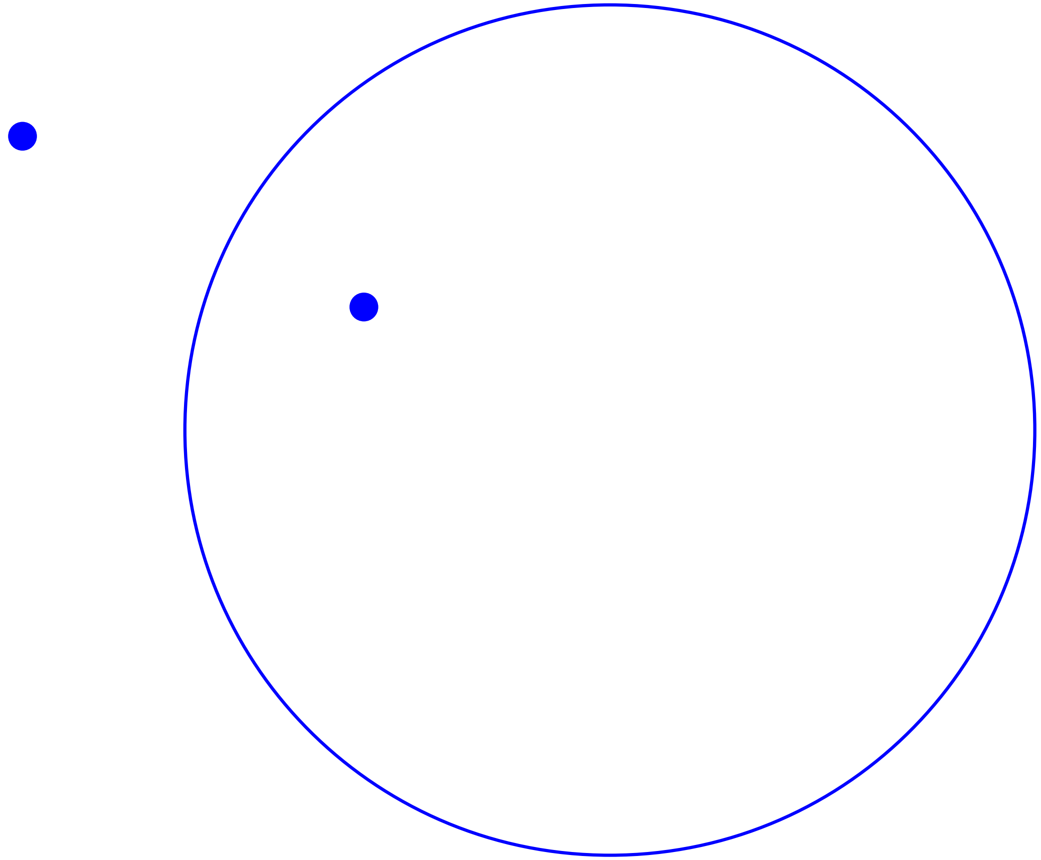
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

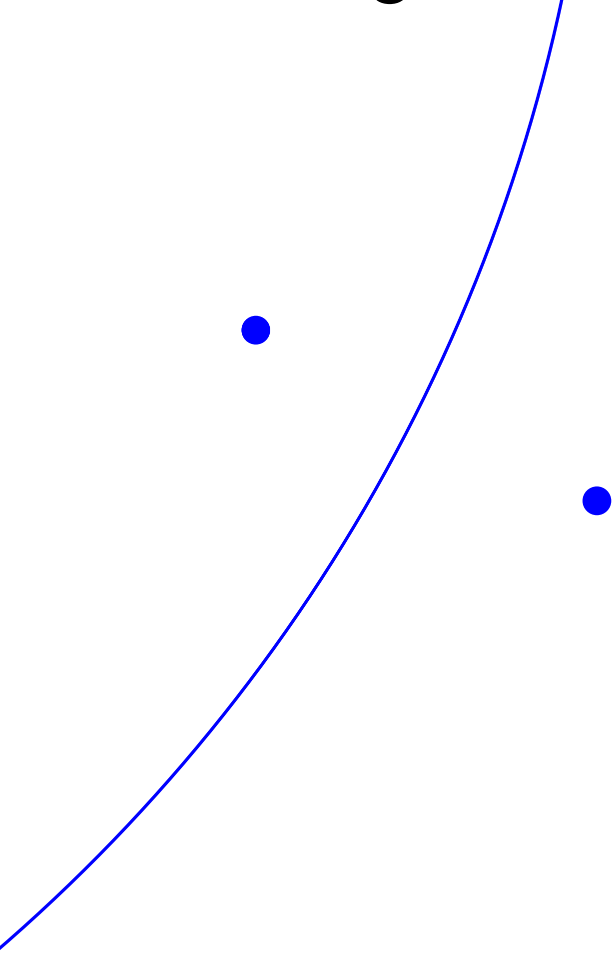
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

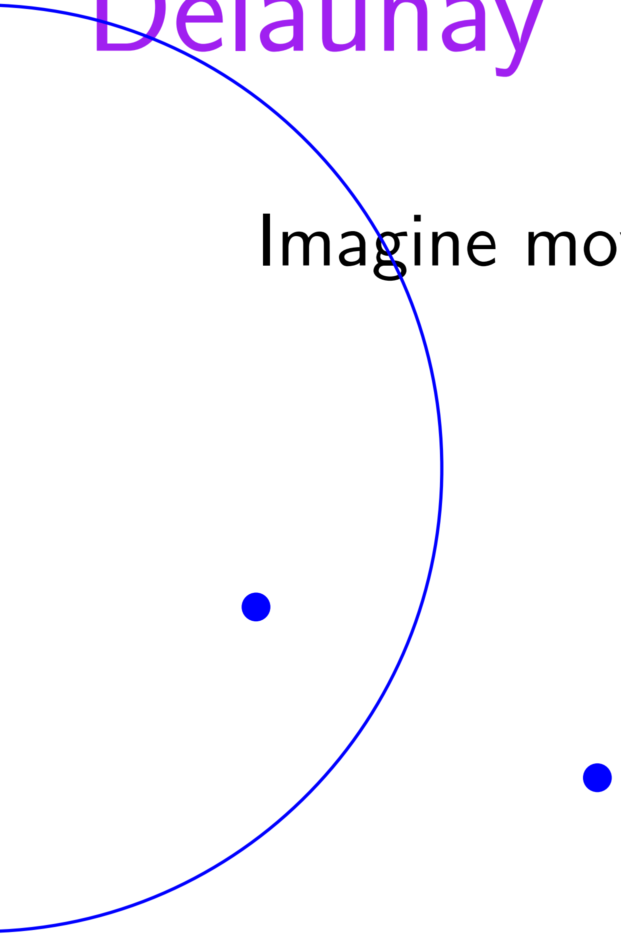
Imagine moving circles



two fixed points

Delaunay Triangulation: pencils of circles

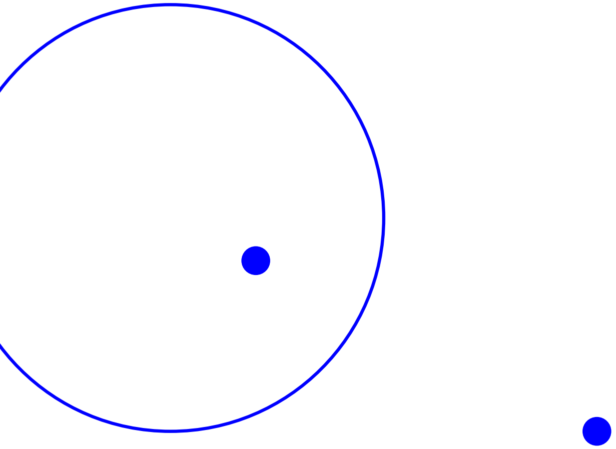
Imagine moving circles



two fixed points

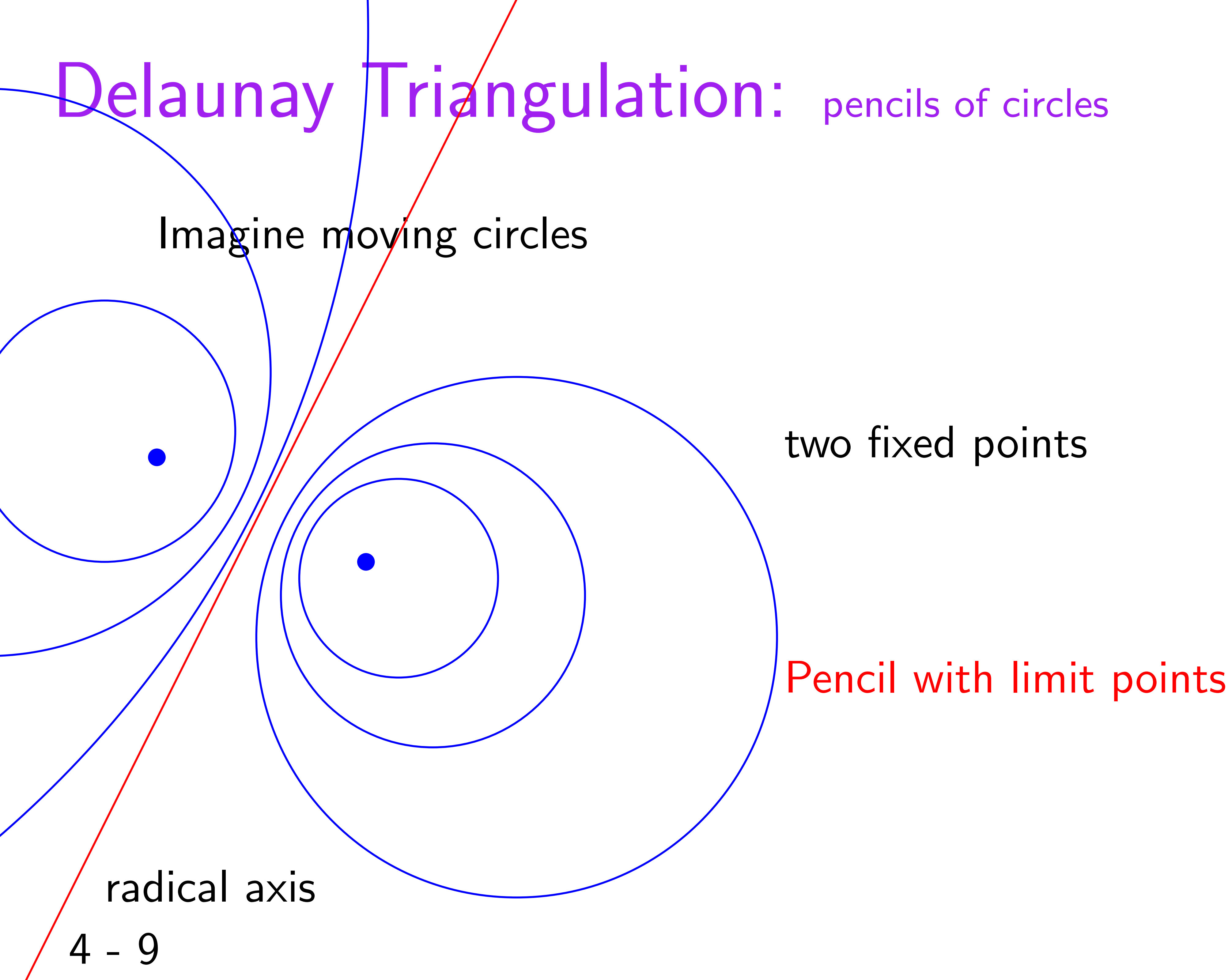
Delaunay Triangulation: pencils of circles

Imagine moving circles

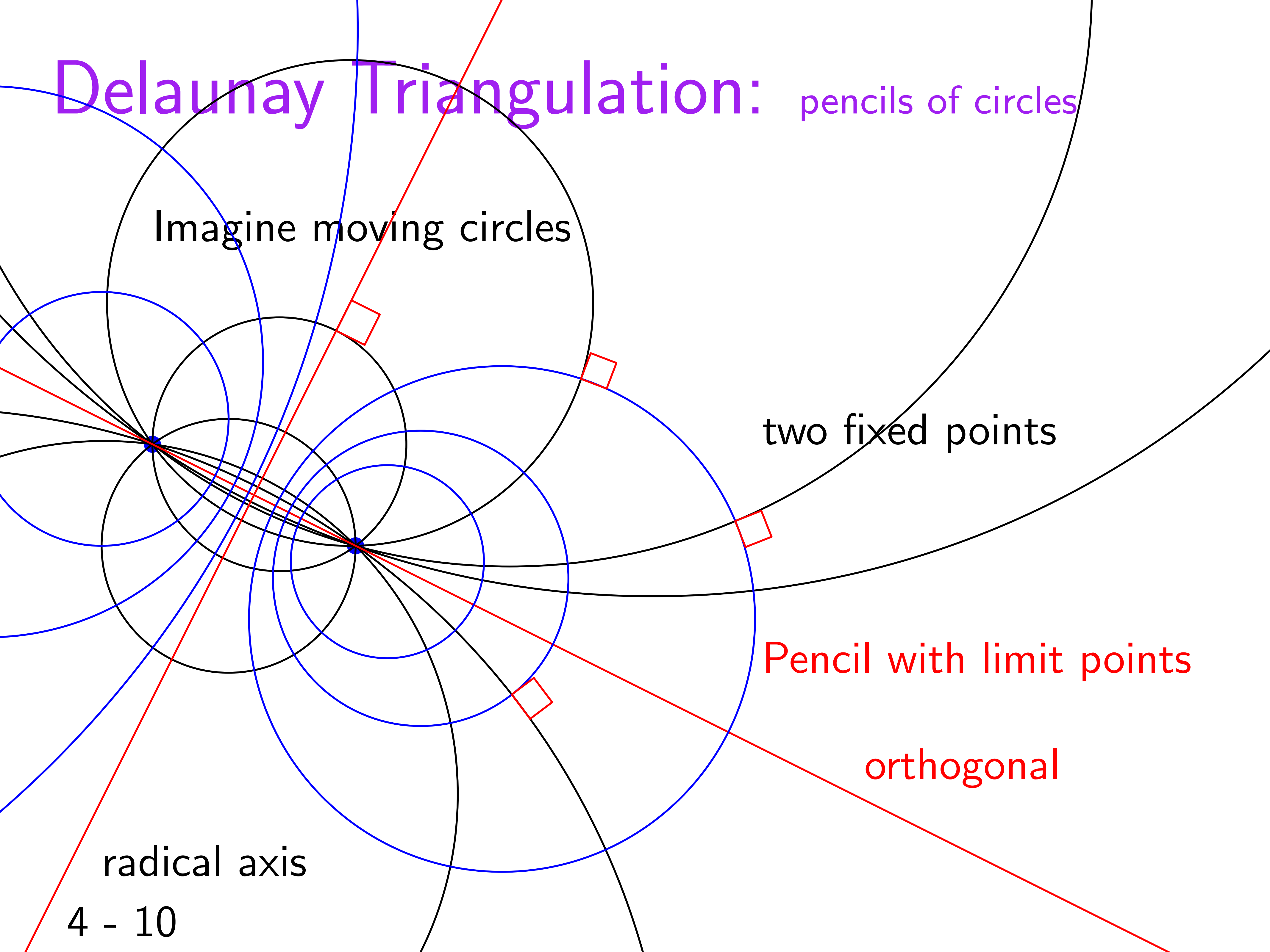


two fixed points

Delaunay Triangulation: pencils of circles



Delaunay Triangulation: pencils of circles



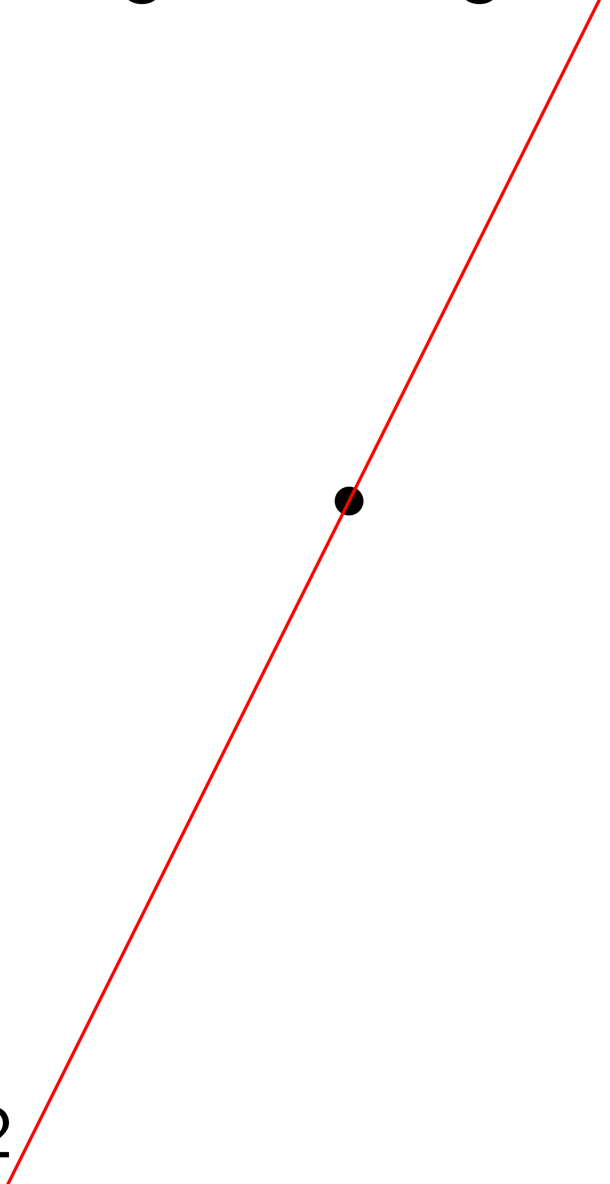
Delaunay Triangulation: pencils of circles

Imagine moving circles

Delaunay Triangulation: pencils of circles

Imagine moving circles

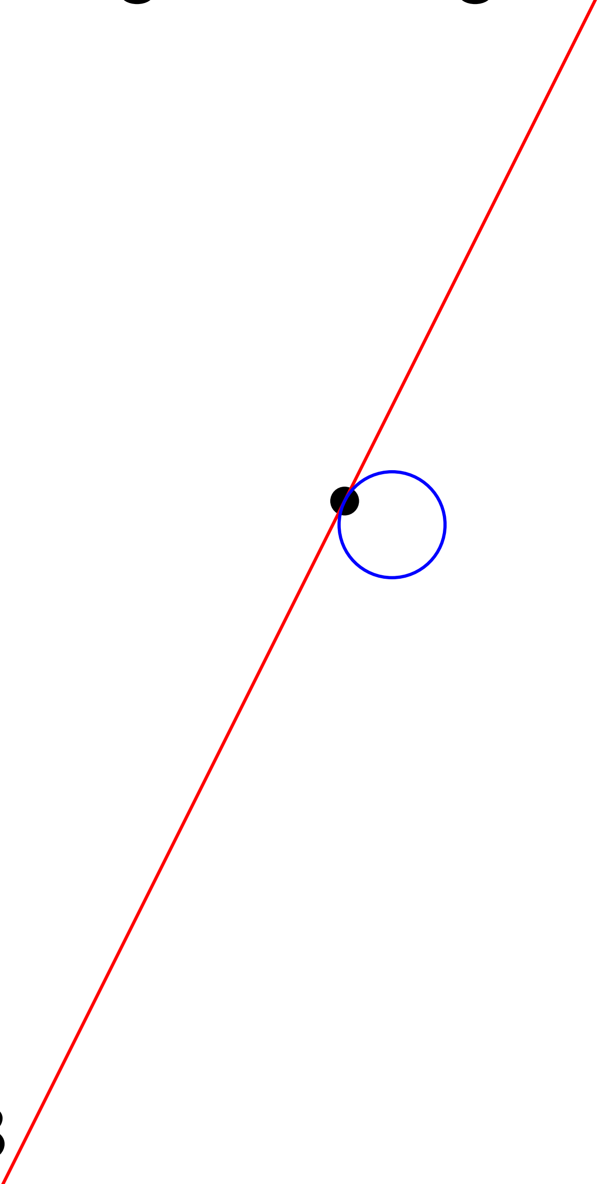
a point on a line



Delaunay Triangulation: pencils of circles

Imagine moving circles

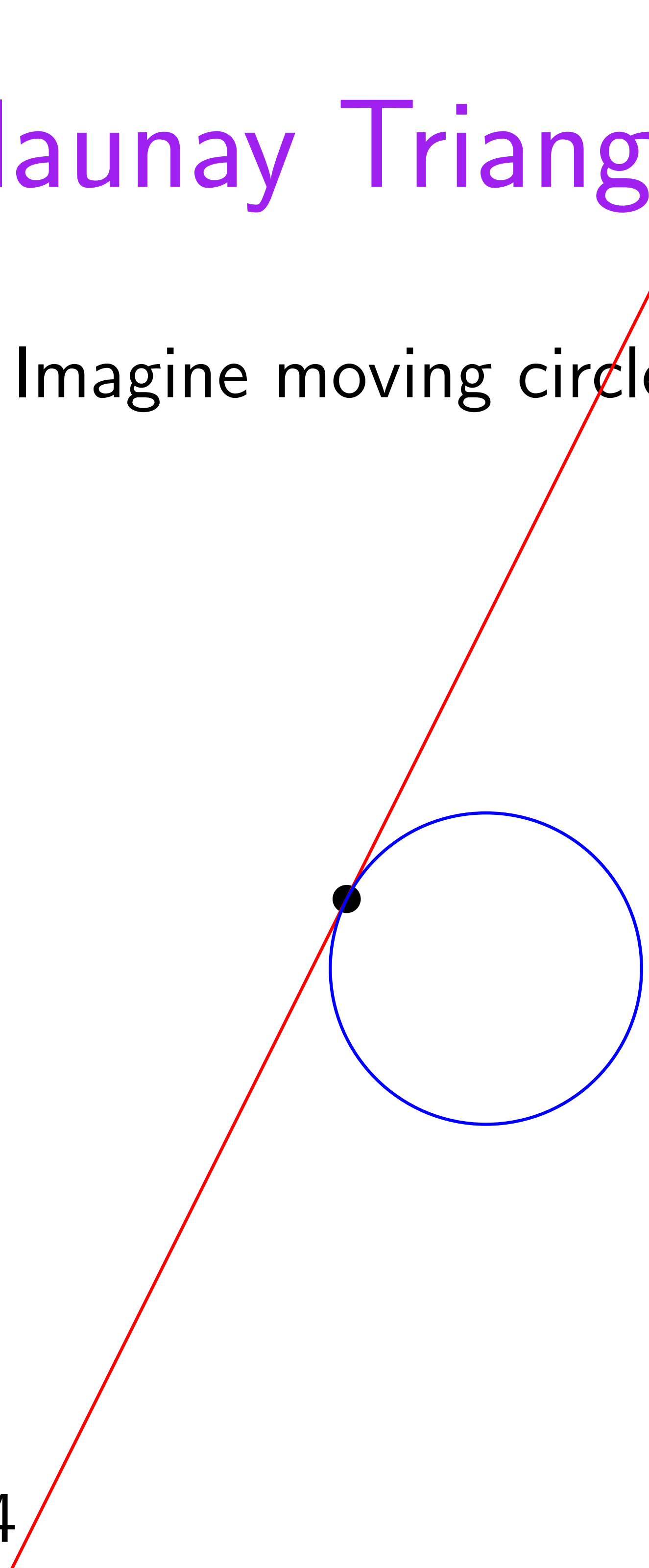
a point on a line



Delaunay Triangulation: pencils of circles

Imagine moving circles

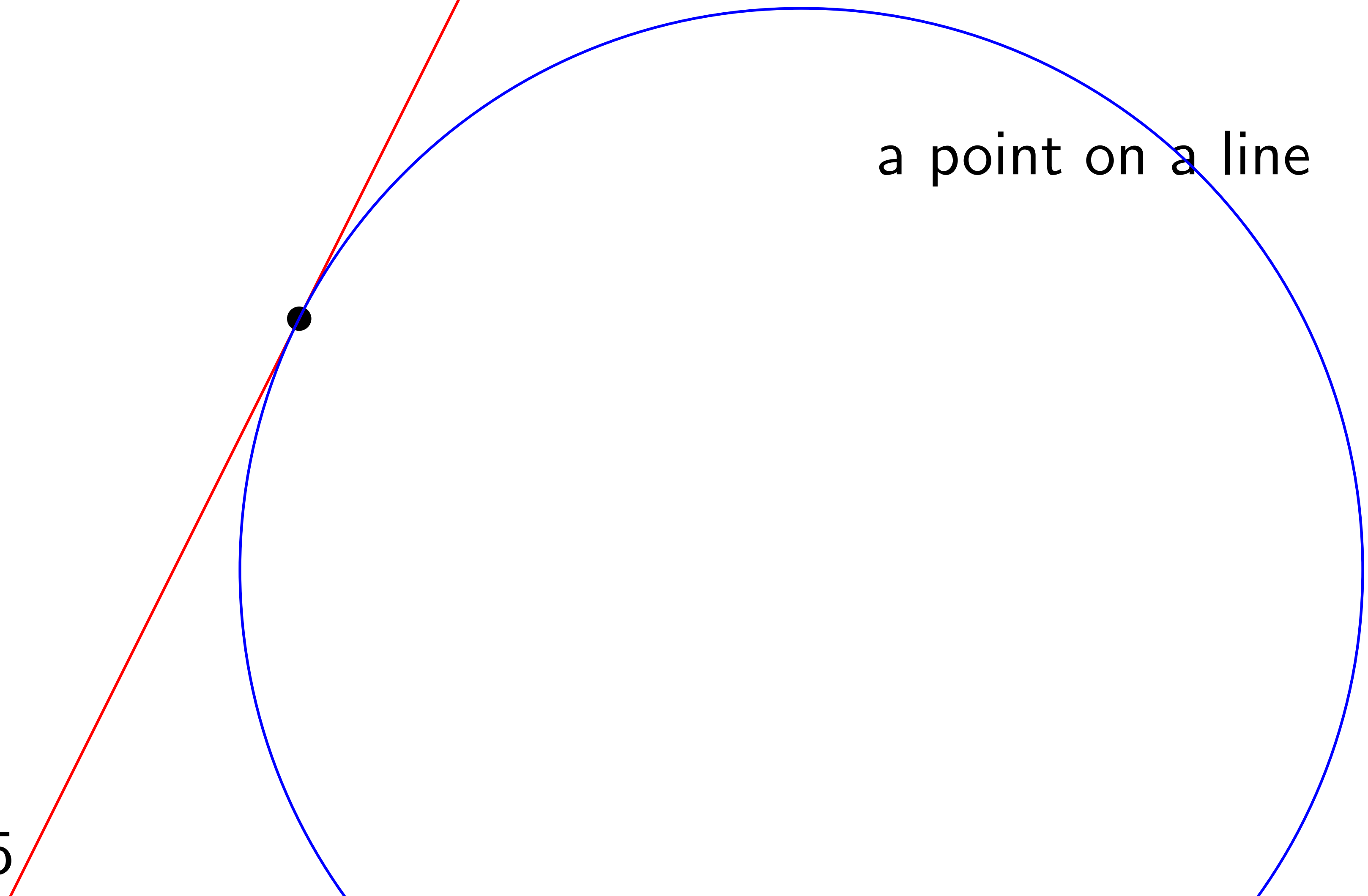
a point on a line



Delaunay Triangulation: pencils of circles

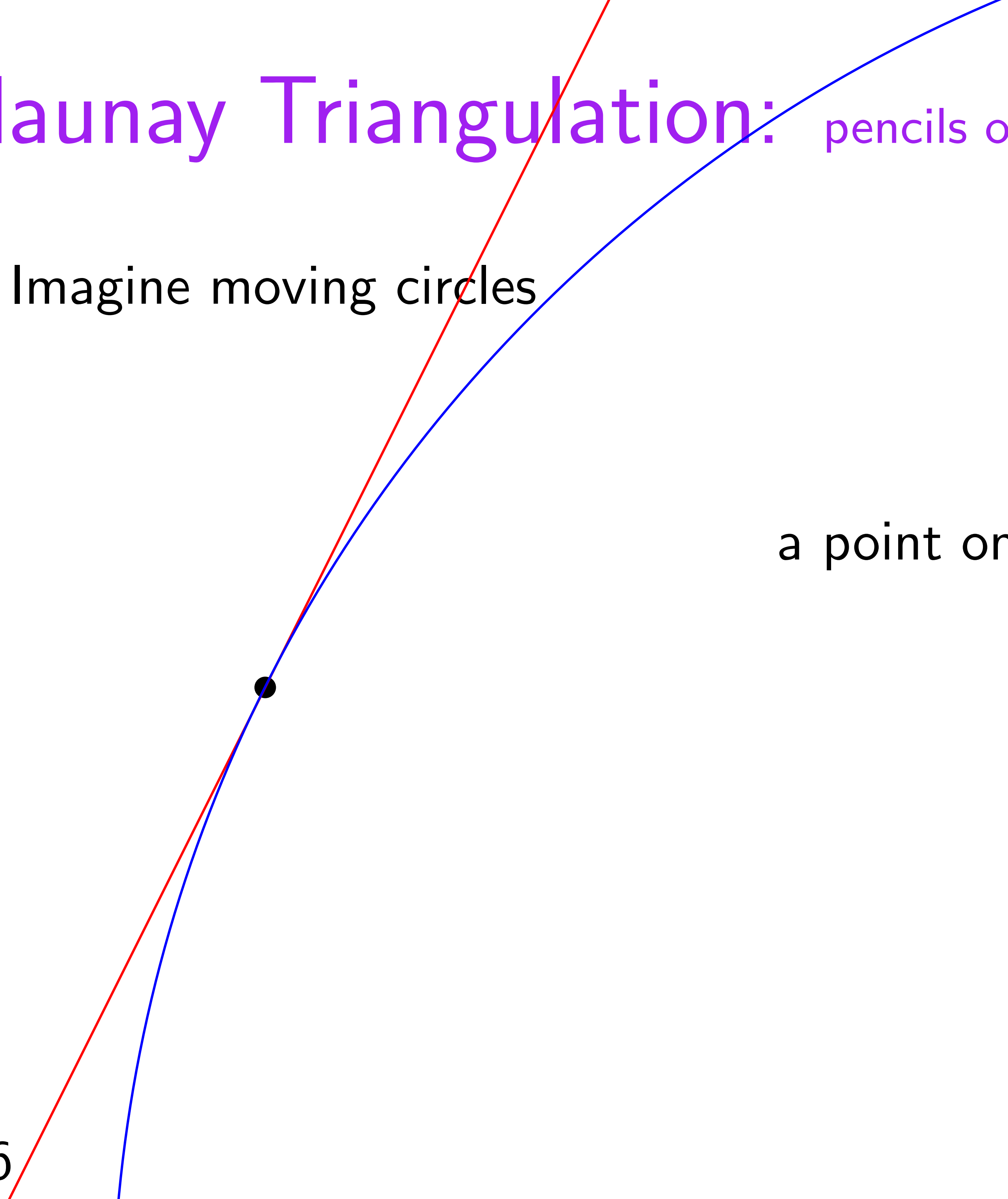
Imagine moving circles

a point on a line



Delaunay Triangulation: pencils of circles

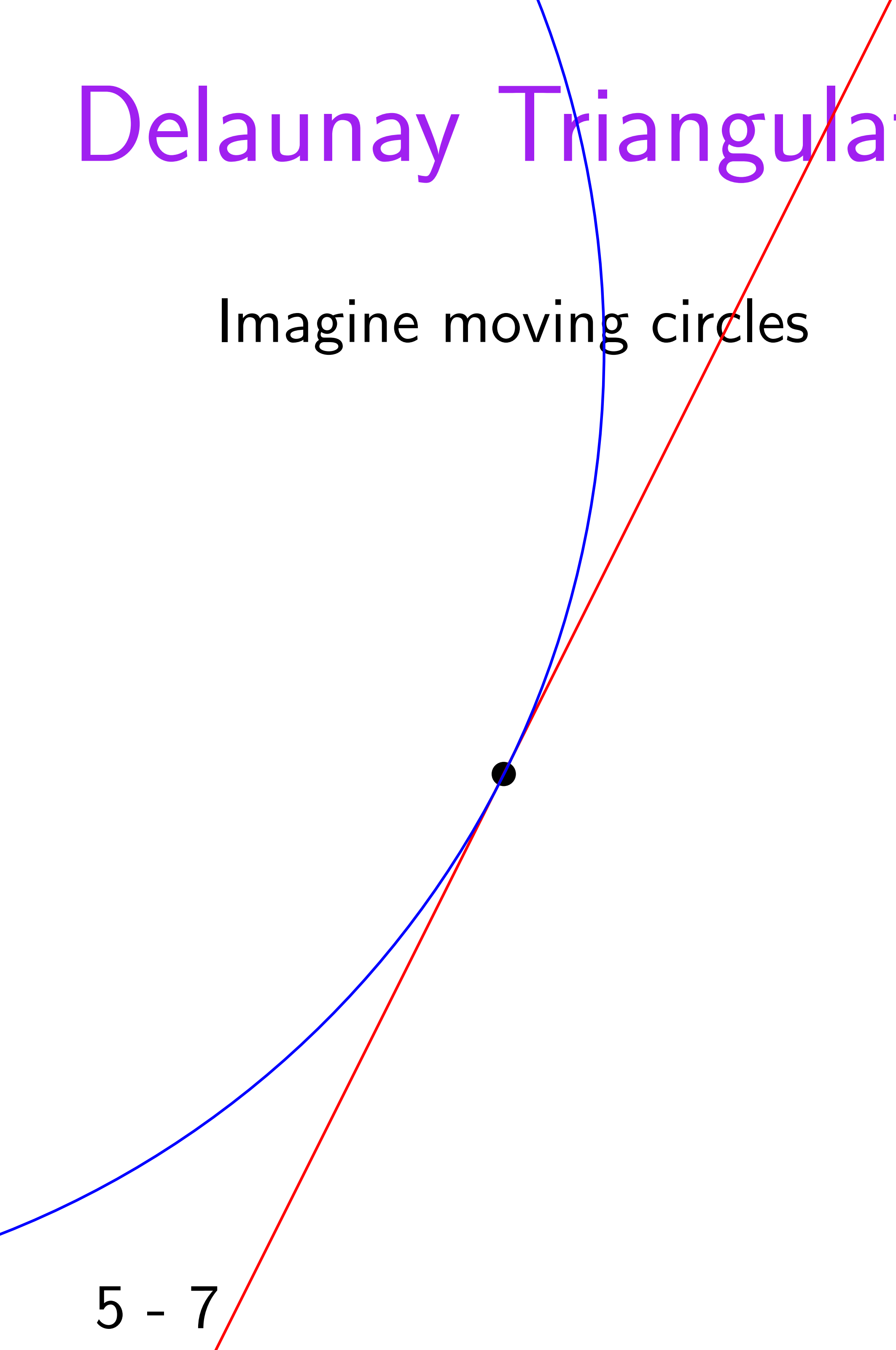
Imagine moving circles



a point on a line

Delaunay Triangulation: pencils of circles

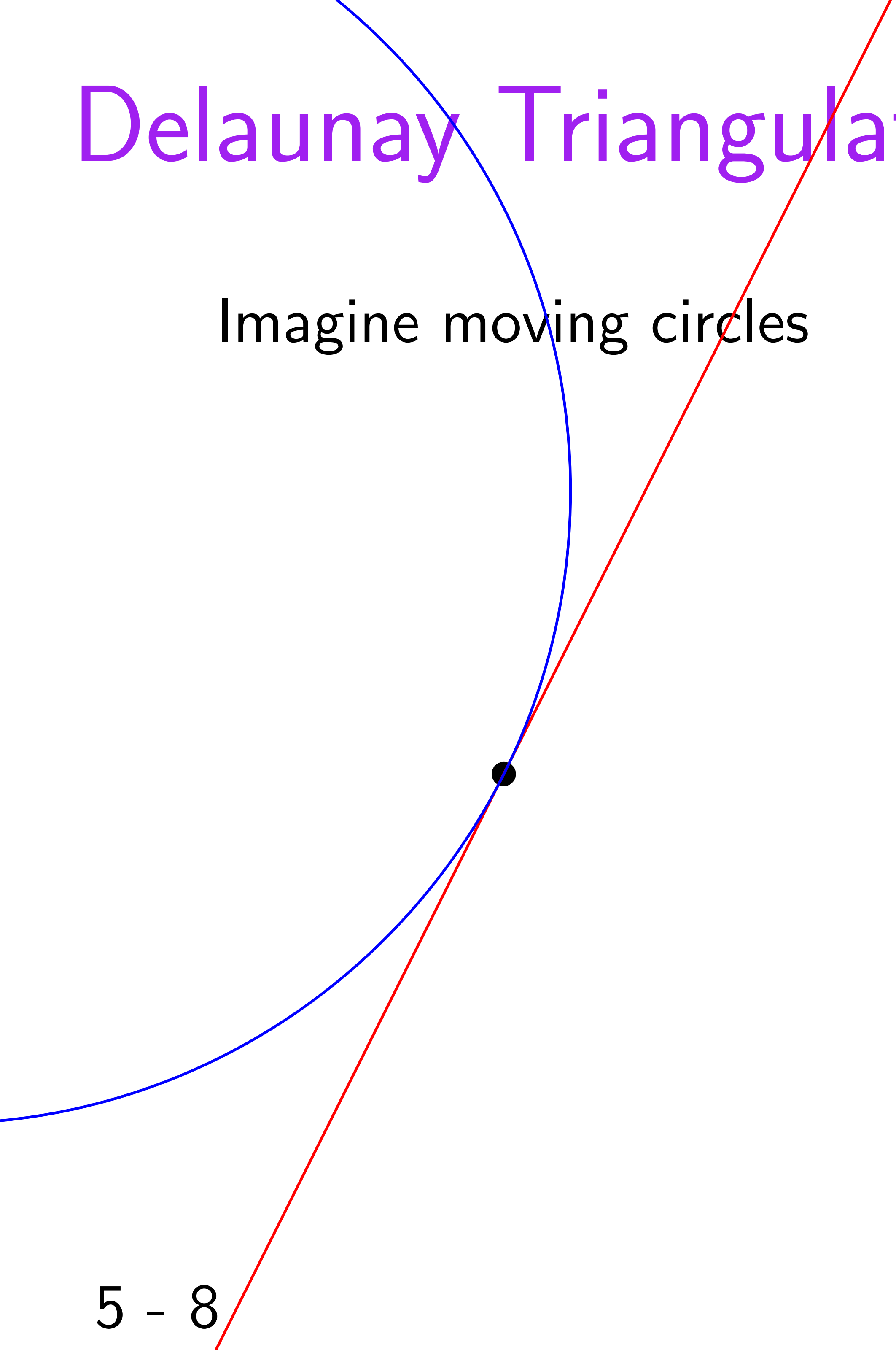
Imagine moving circles



a point on a line

Delaunay Triangulation: pencils of circles

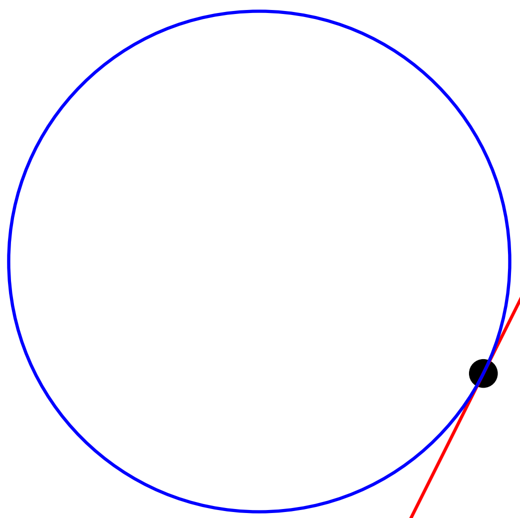
Imagine moving circles



a point on a line

Delaunay Triangulation: pencils of circles

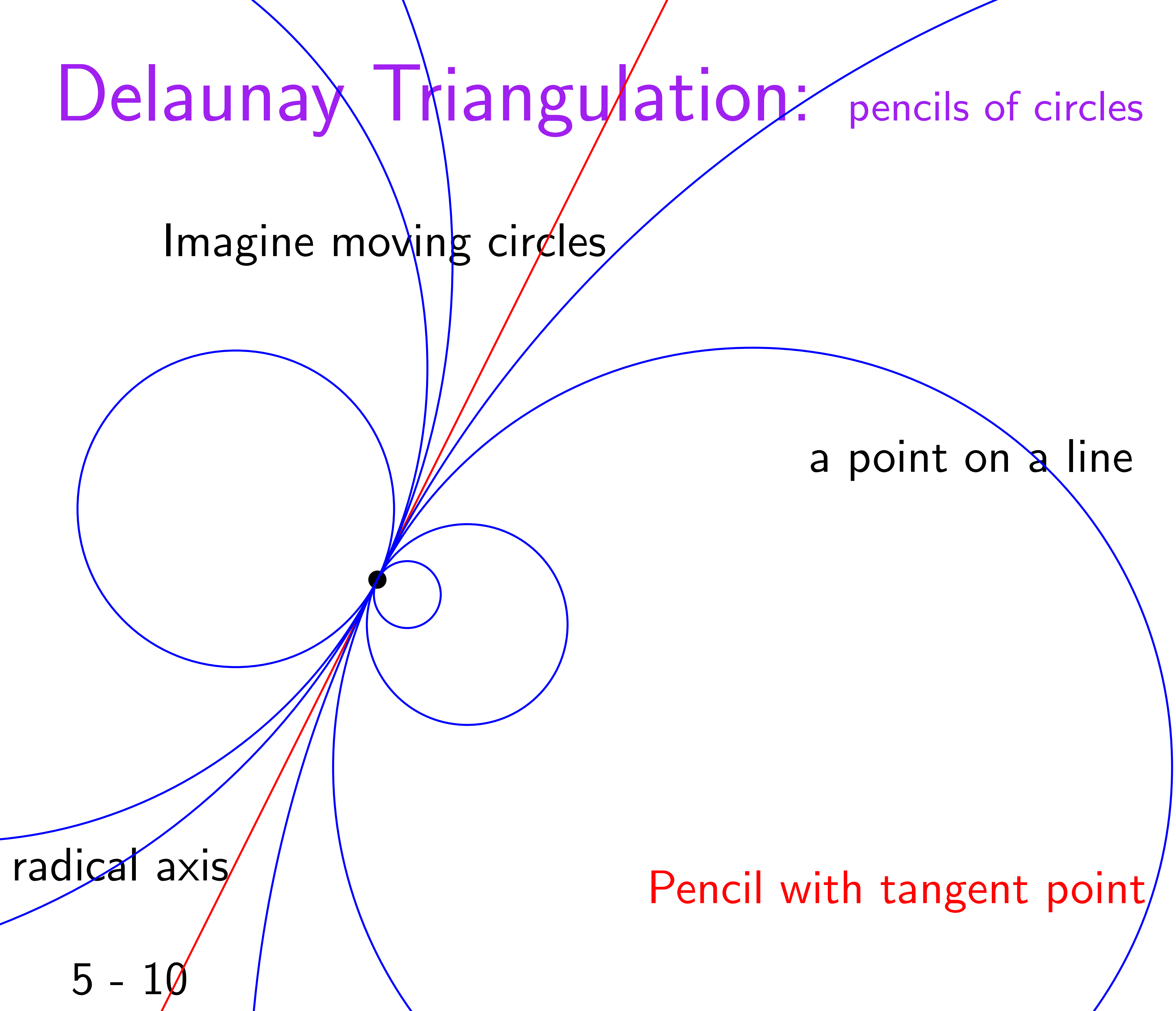
Imagine moving circles



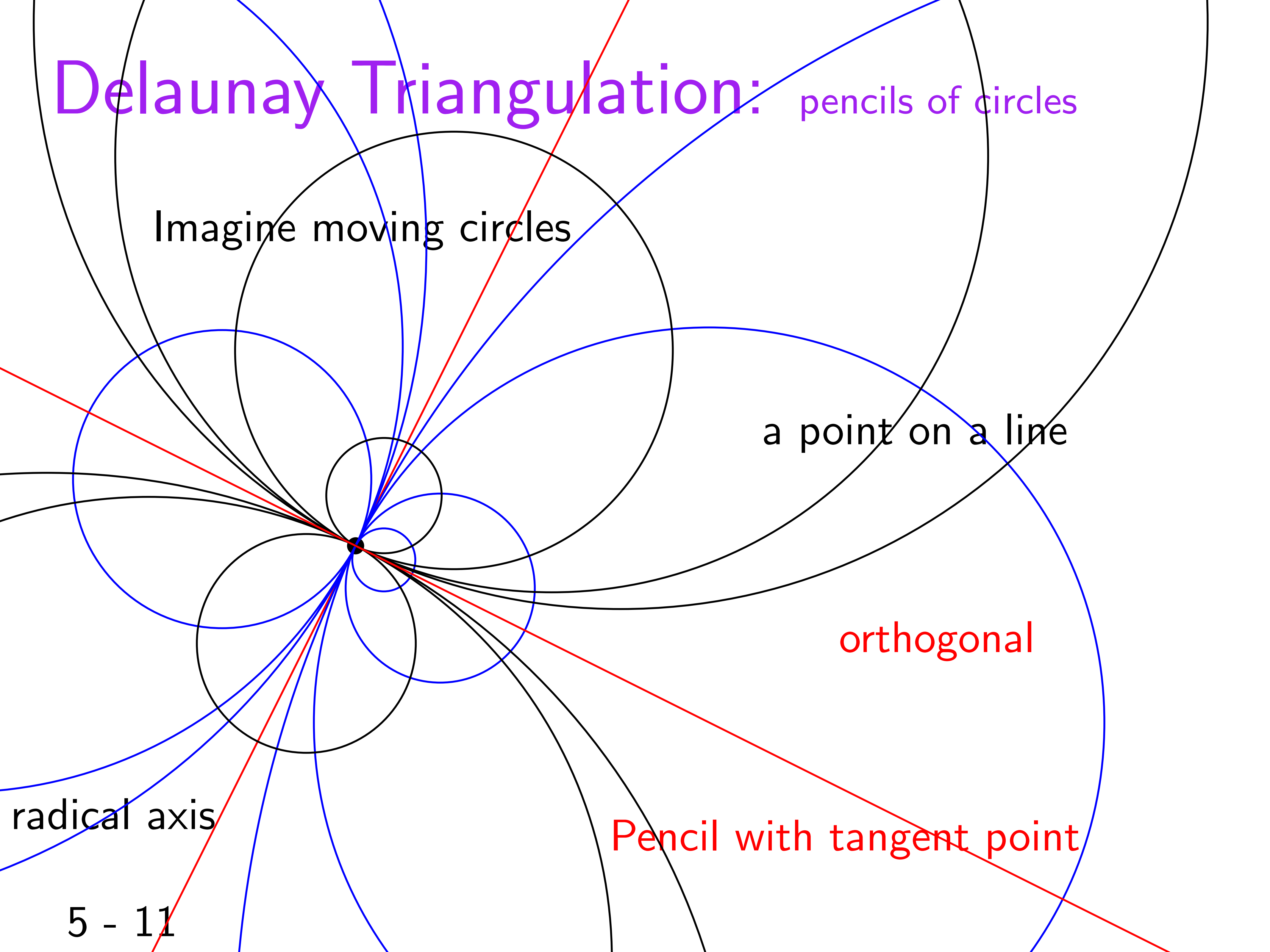
a point on a line

Delaunay Triangulation: pencils of circles

Imagine moving circles



Delaunay Triangulation: pencils of circles



Delaunay Triangulation: pencils of circles

Circle equation

$$x^2 + y^2 - 2ax - 2by + c = 0$$

Delaunay Triangulation: pencils of circles

Circle equation

$$x^2 + y^2 - 2ax - 2by + c = 0$$

Another circle equation

$$x^2 + y^2 - 2a'x - 2b'y + c' = 0$$

Delaunay Triangulation: pencils of circles

Circle equation

$$x^2 + y^2 - 2ax - 2by + c = 0$$

Another circle equation

$$x^2 + y^2 - 2a'x - 2b'y + c' = 0$$

Pencil of circles

$$\lambda \cdot (x^2 + y^2 - 2ax - 2by + c)$$
$$+(1-\lambda) \cdot (x^2 + y^2 - 2a'x - 2b'y + c') = 0$$

Delaunay Triangulation: pencils of circles

Circle equation

$$x^2 + y^2 - 2ax - 2by + c = 0$$

Another circle equation

$$x^2 + y^2 - 2a'x - 2b'y + c' = 0$$

Pencil of circles

$$\lambda \cdot (x^2 + y^2 - 2ax - 2by + c)$$
$$+(1-\lambda) \cdot (x^2 + y^2 - 2a'x - 2b'y + c') = 0$$

A special "circle: the radical axis

$$\lambda = \infty$$

Delaunay Triangulation: pencils of circles

Power of a point w.r.t a circle

$$x^2 + y^2 - 2ax - 2by + c$$

Delaunay Triangulation: pencils of circles

Power of a point w.r.t a circle

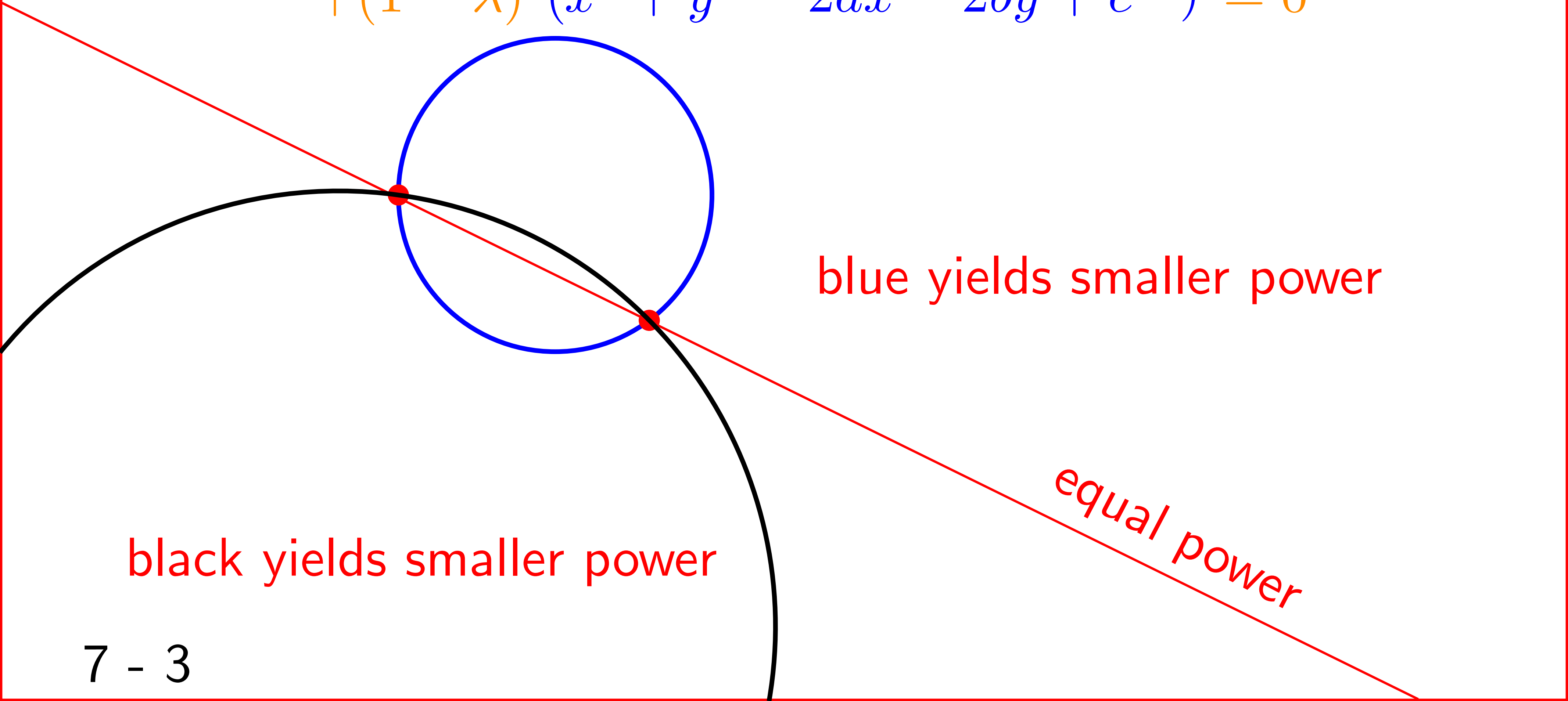
$$x^2 + y^2 - 2ax - 2by + c$$

$= 0$	on the circle
< 0	inside the circle
> 0	outside the circle

Delaunay Triangulation: pencils of circles

Power of a point w.r.t a circle

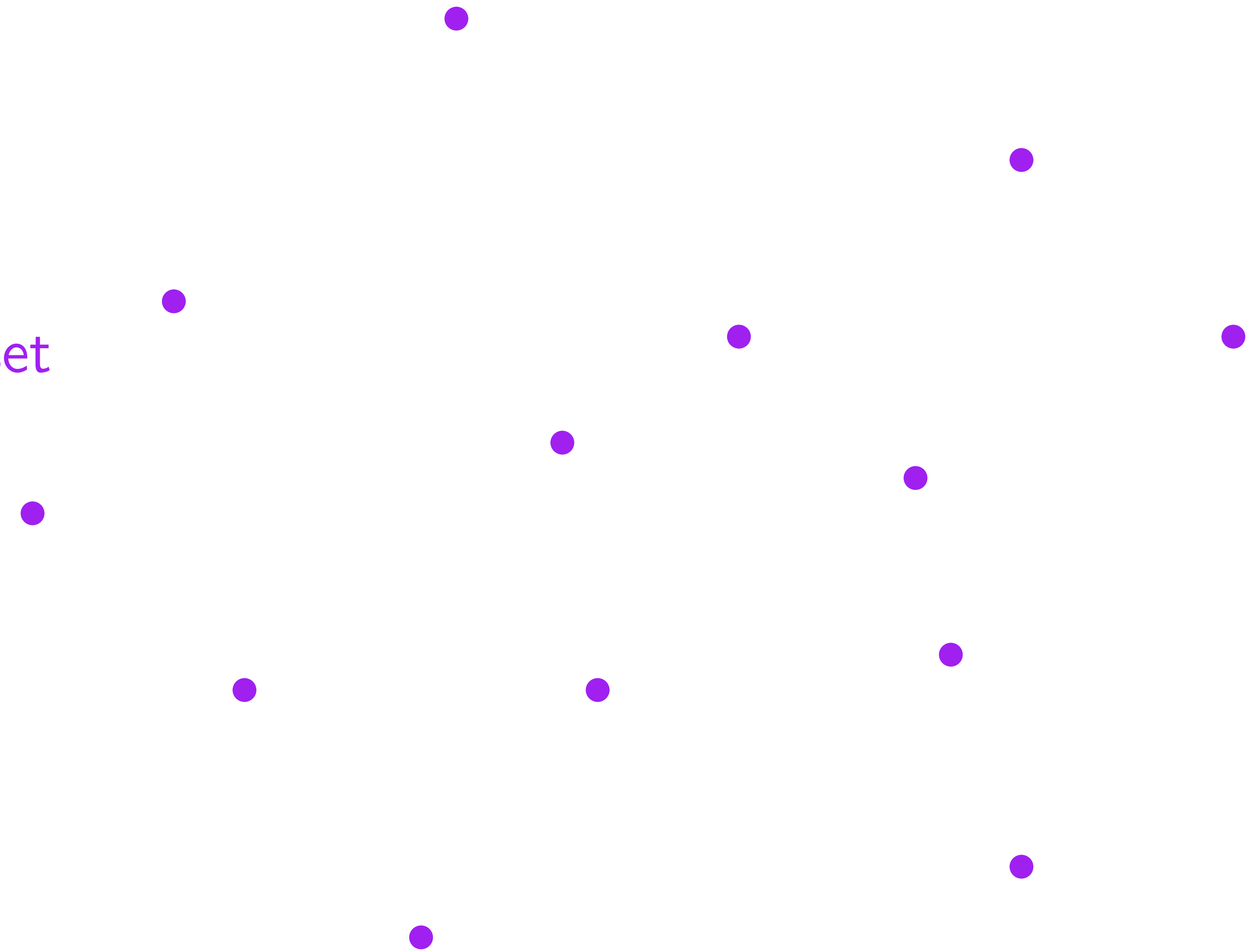
$$\lambda \quad (x^2 + y^2 - 2a'x - 2b'y + c')$$
$$+(1-\lambda)\ (x^2 + y^2 - 2ax - 2by + c\quad) = 0$$



Delaunay Triangulation: definition, empty circle property

Delaunay Triangulation: definition, empty circle property

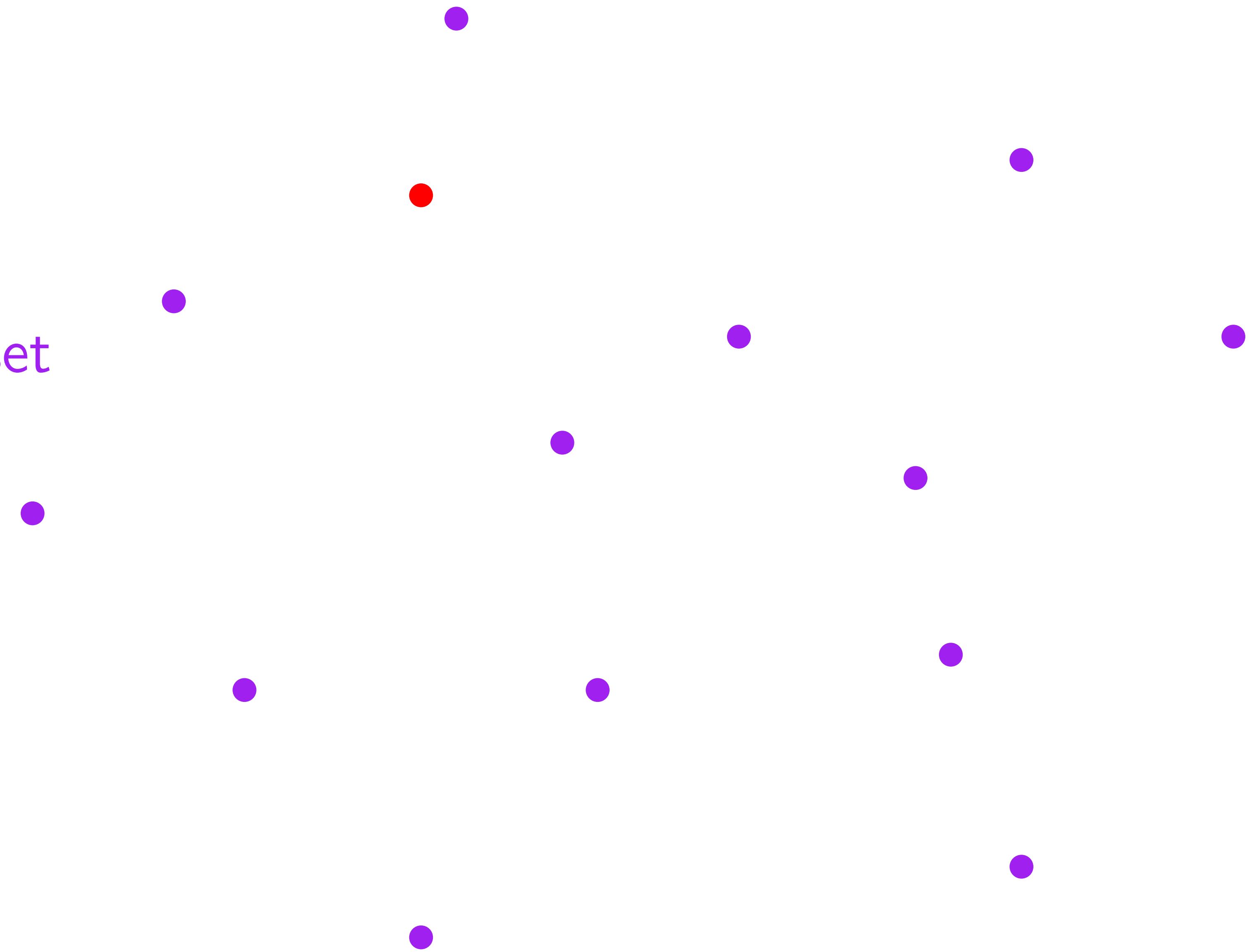
Point set



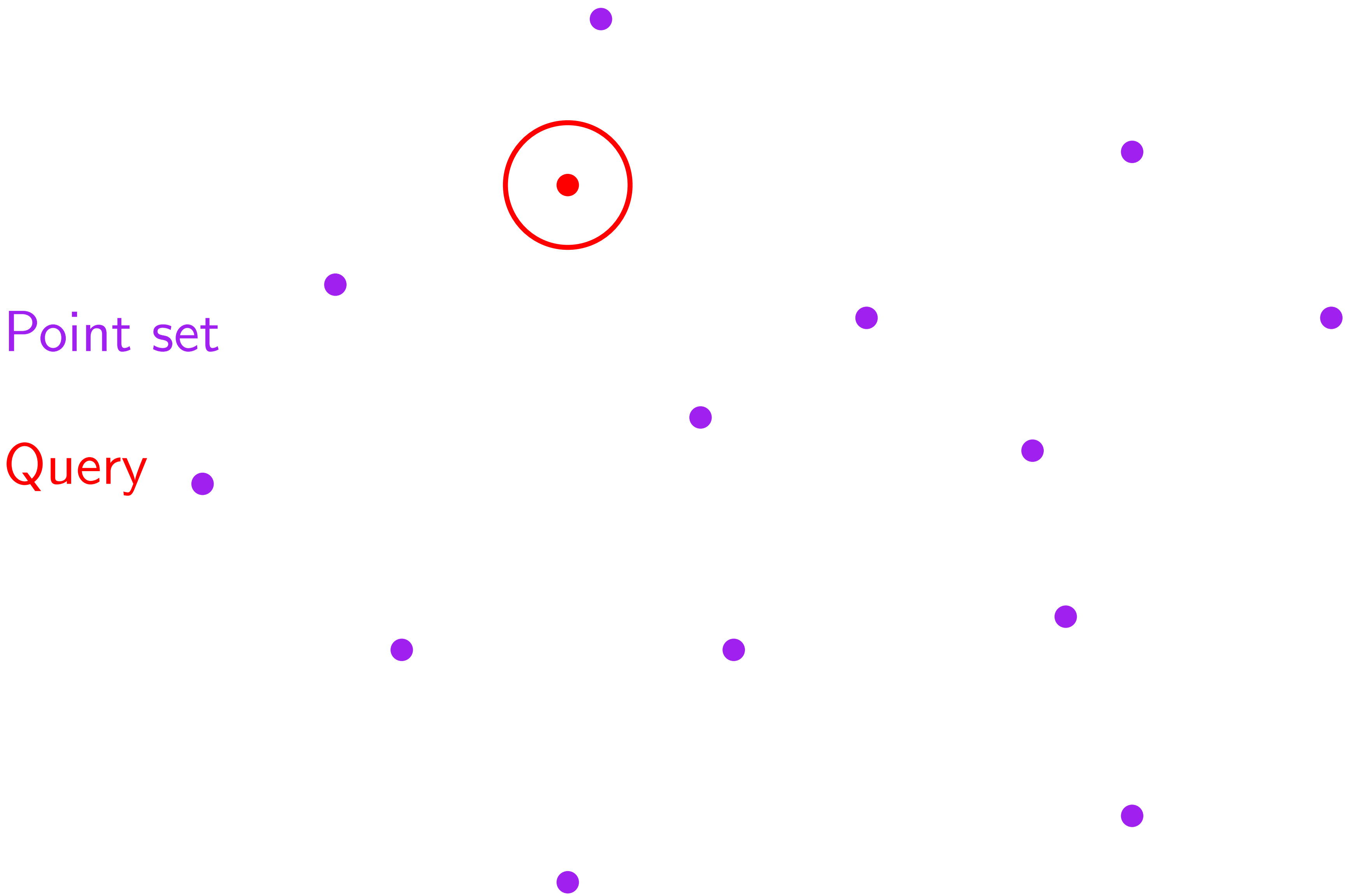
Delaunay Triangulation: definition, empty circle property

Point set

Query



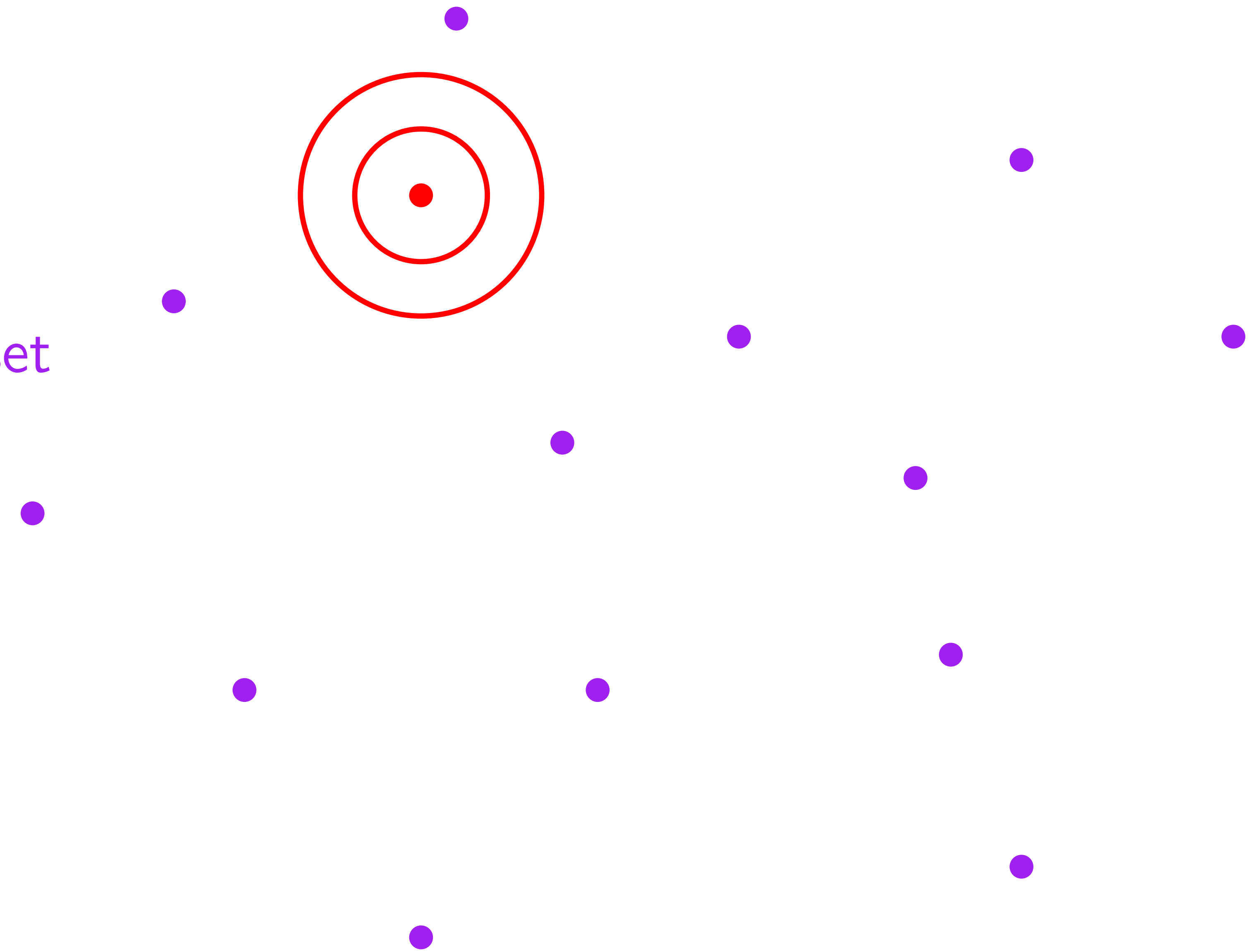
Delaunay Triangulation: definition, empty circle property



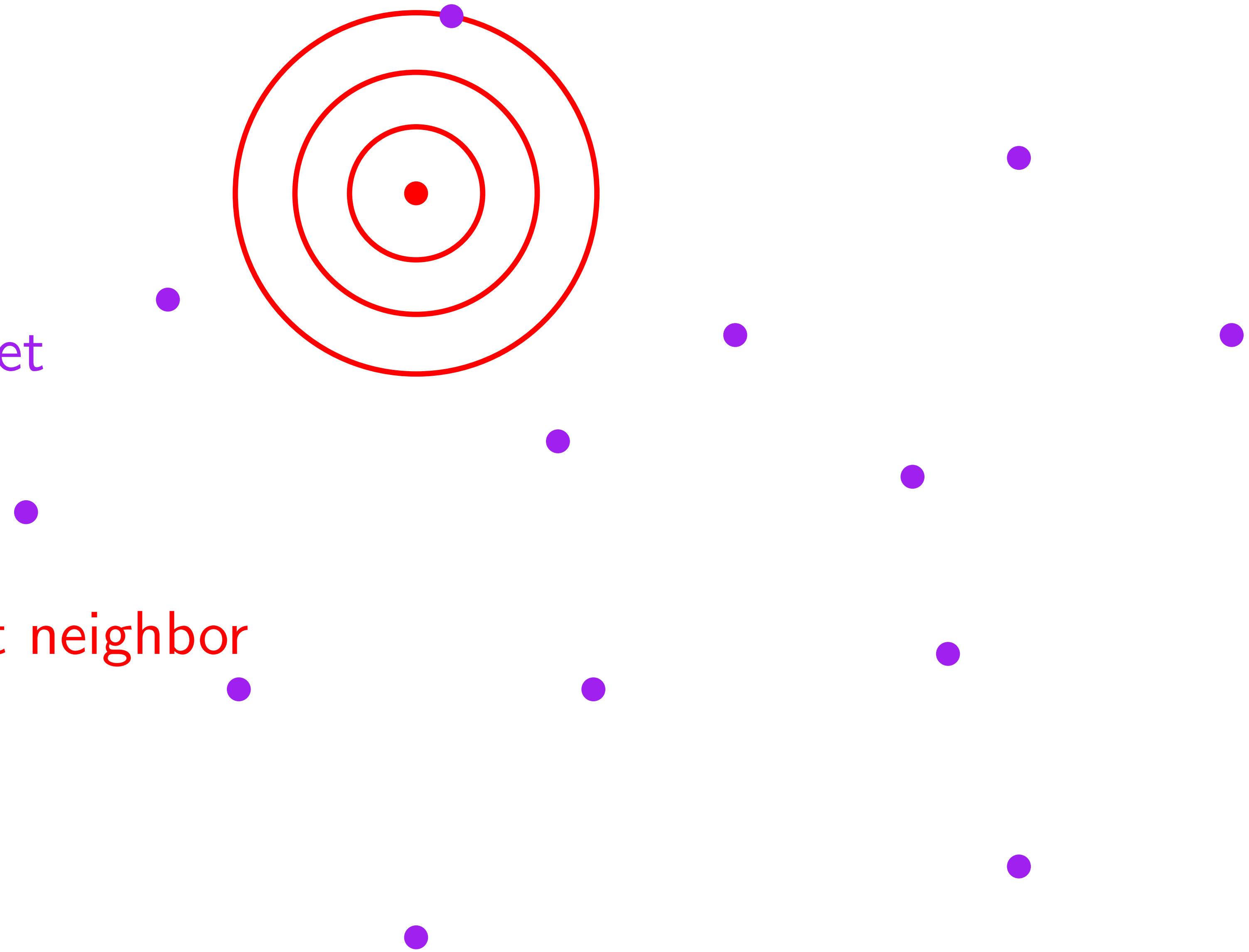
Delaunay Triangulation: definition, empty circle property

Point set

Query



Delaunay Triangulation: definition, empty circle property

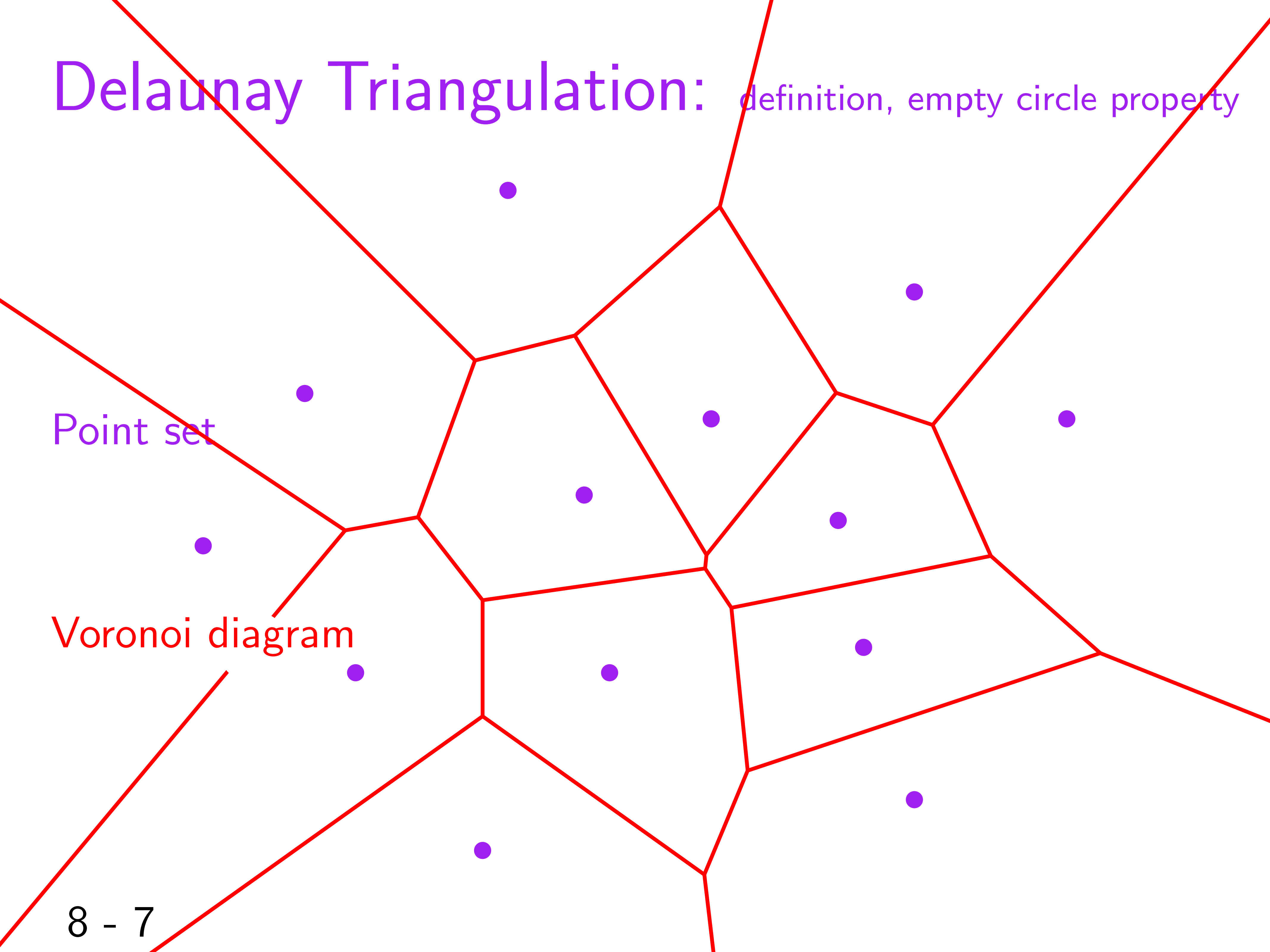


Point set

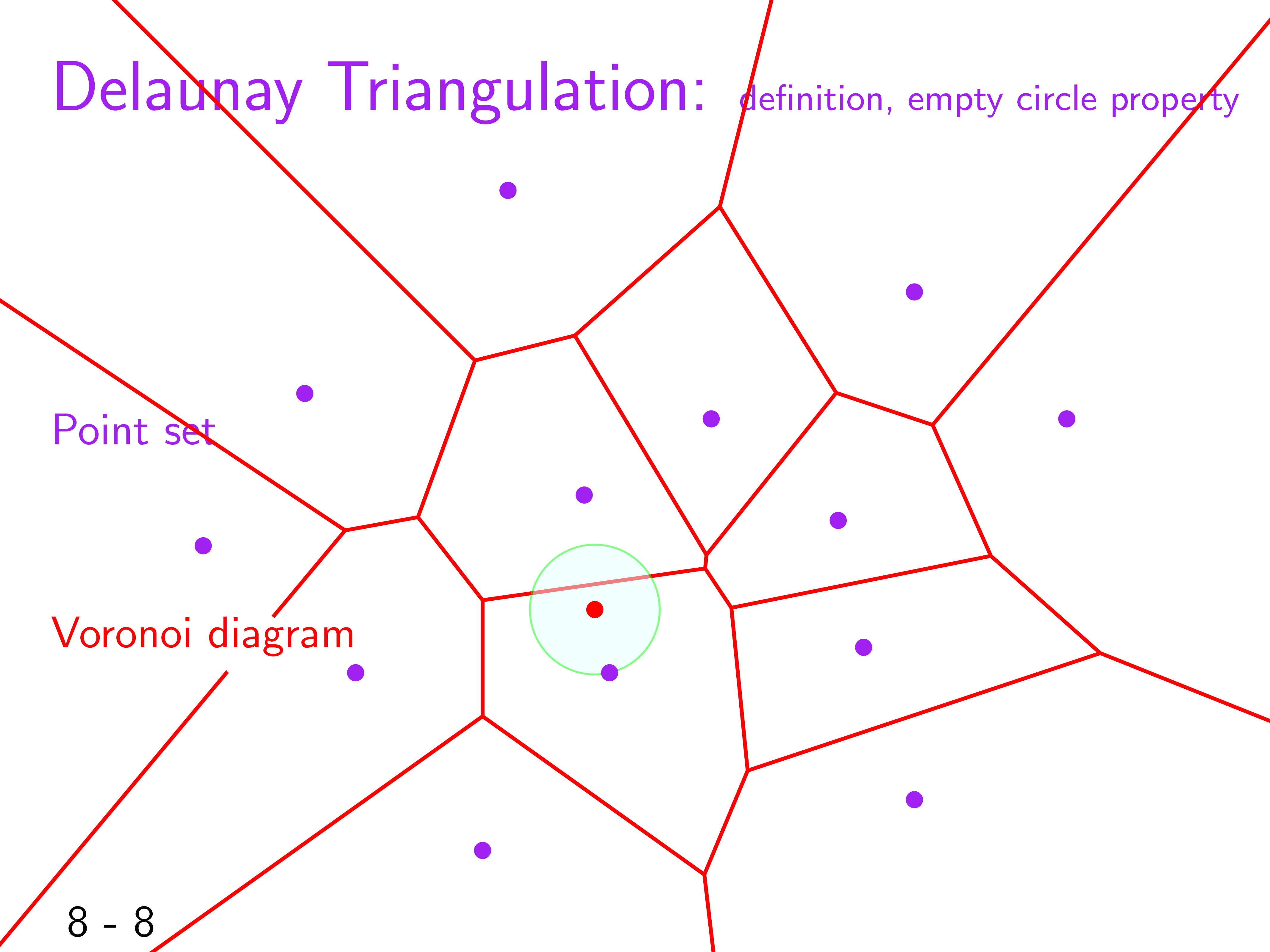
Query

Nearest neighbor

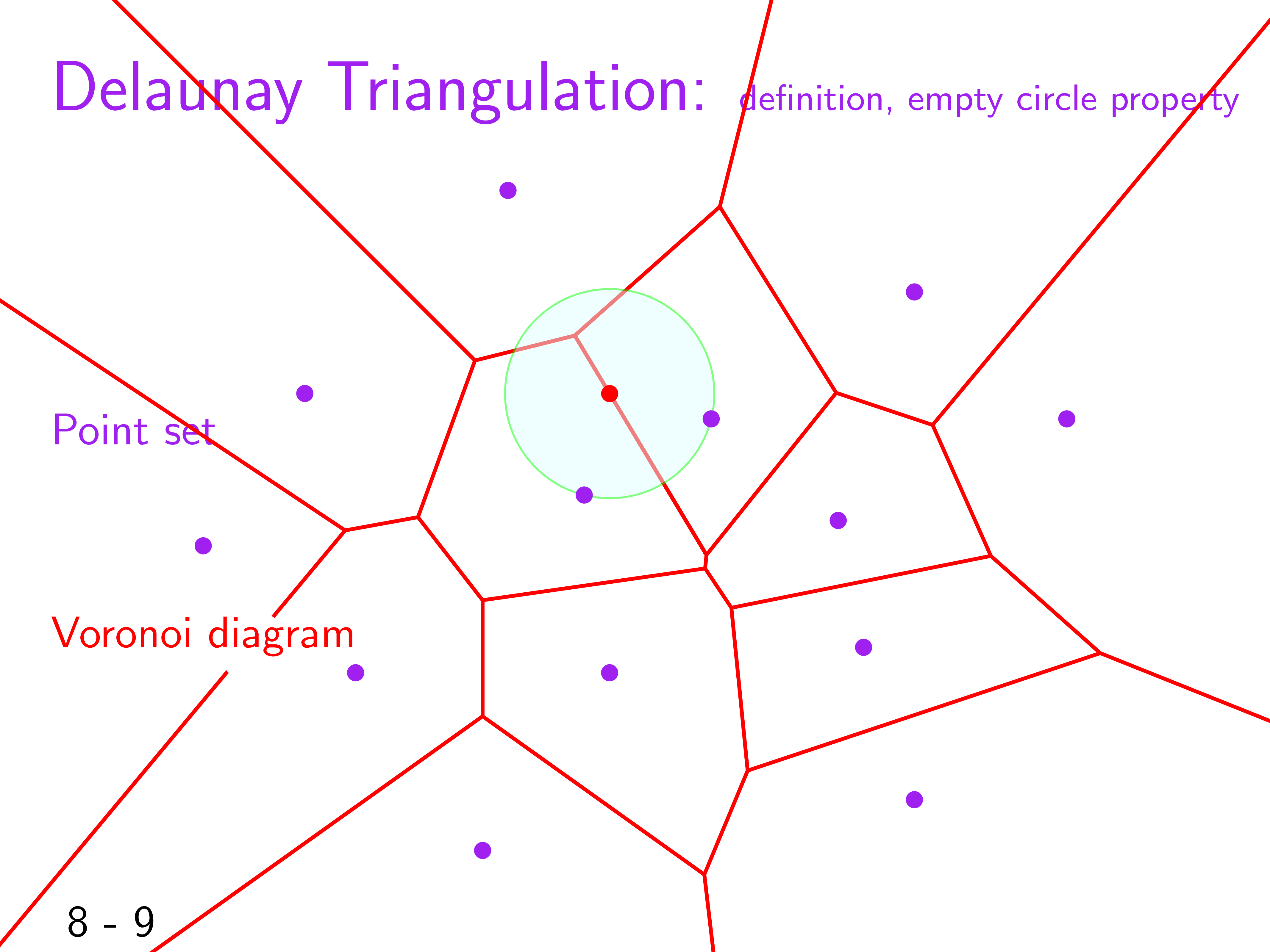
Delaunay Triangulation: definition, empty circle property



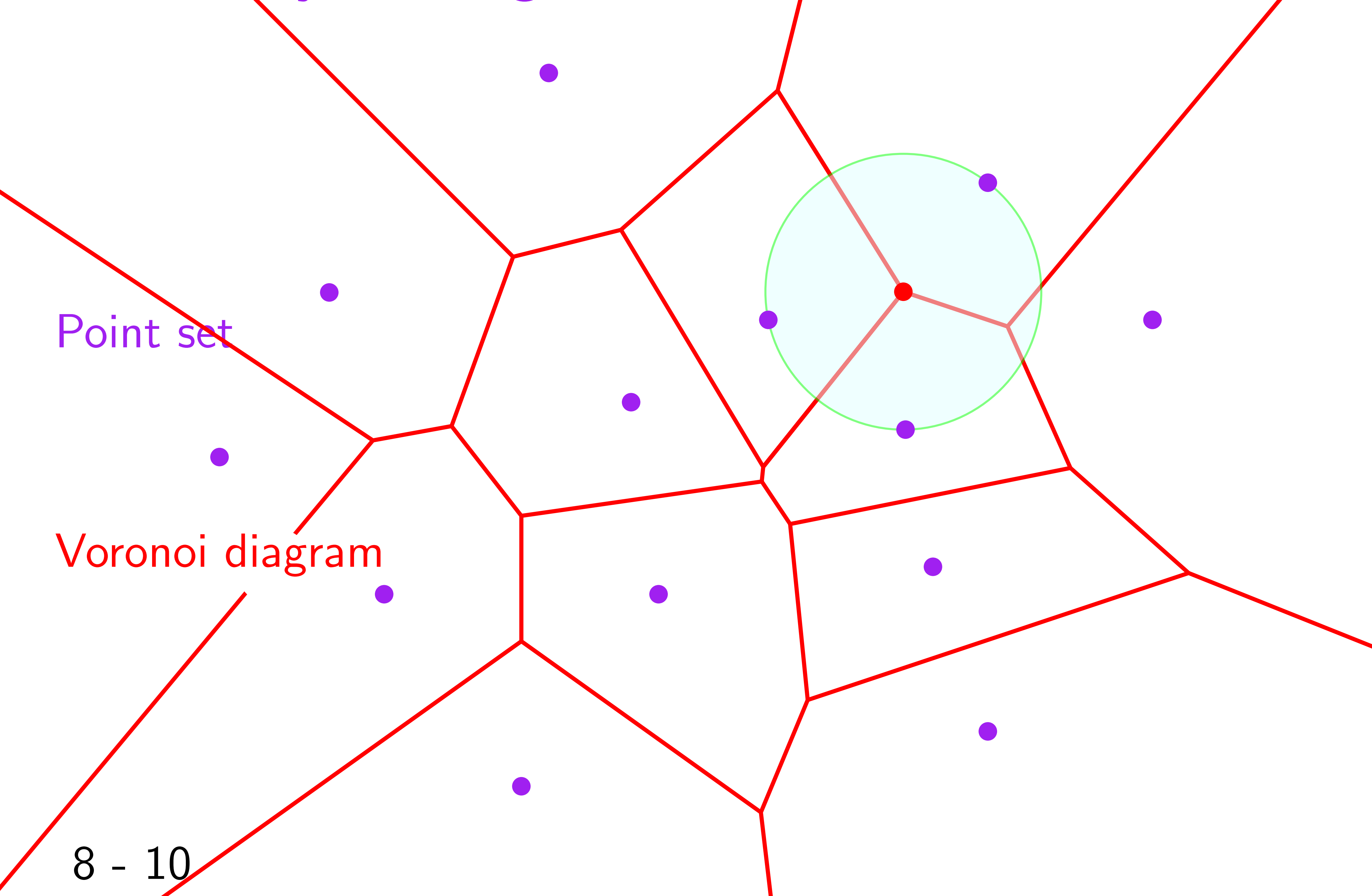
Delaunay Triangulation: definition, empty circle property



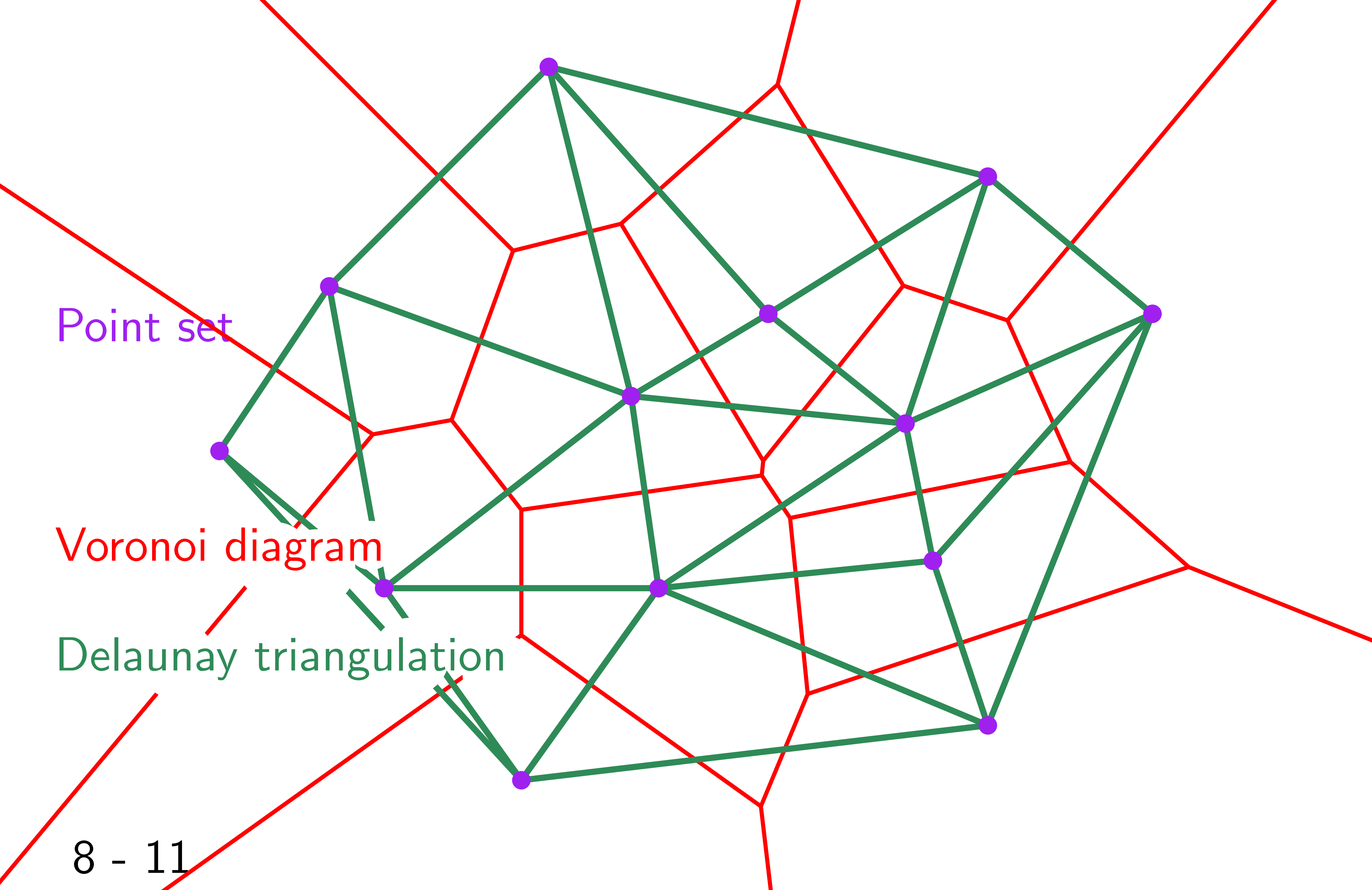
Delaunay Triangulation: definition, empty circle property



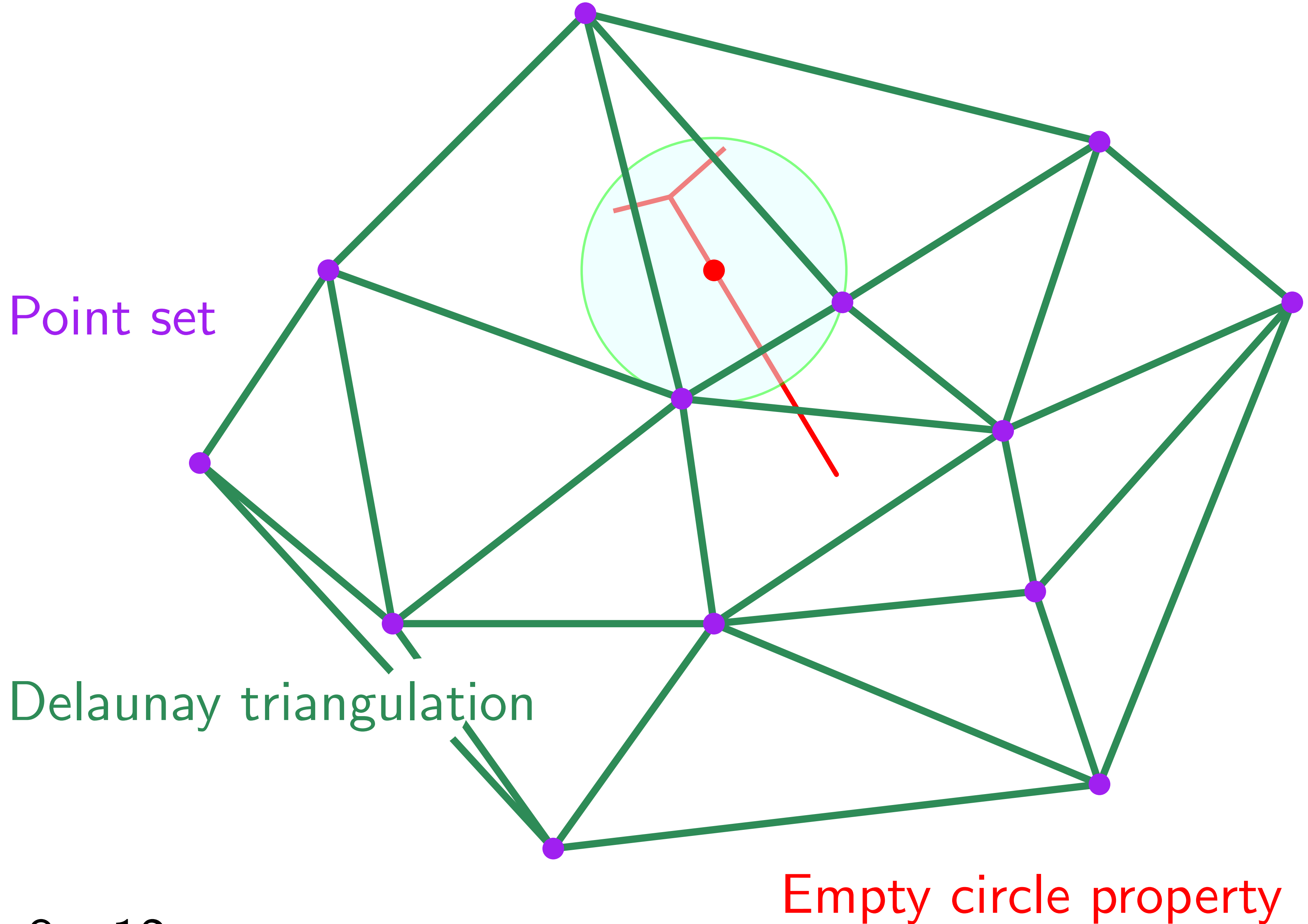
Delaunay Triangulation: definition, empty circle property



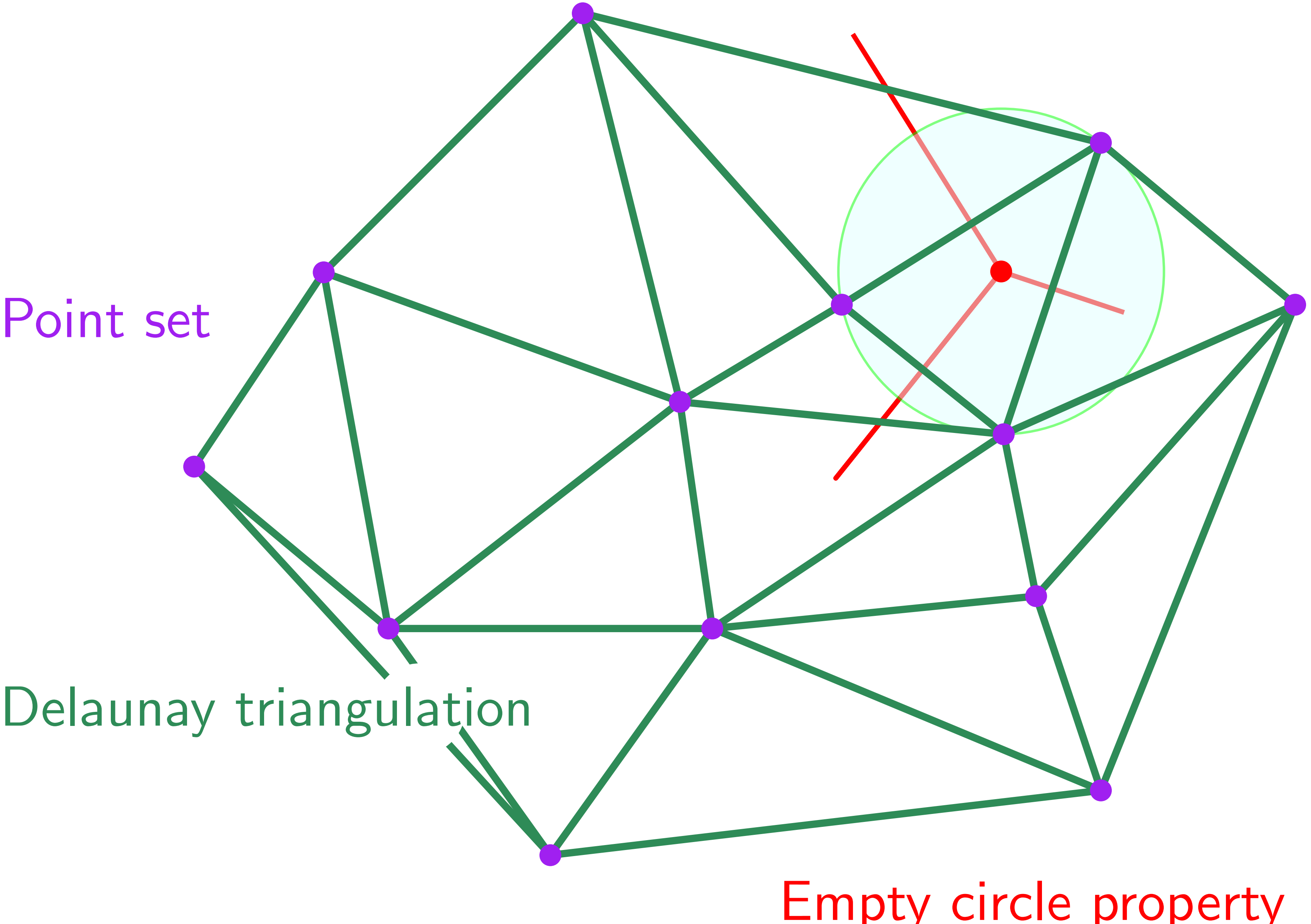
Delaunay Triangulation: definition, empty circle property



Delaunay Triangulation: definition, empty circle property



Delaunay Triangulation: definition, empty circle property



Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown curve

Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown curve

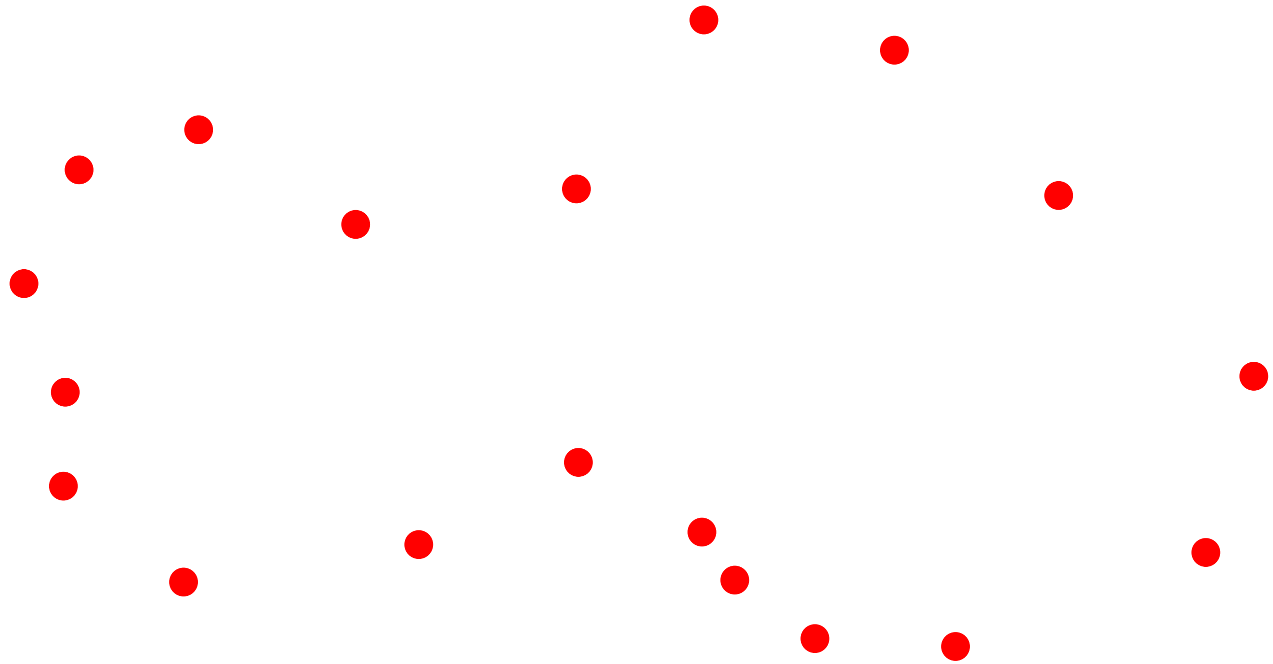
Output: the curve (the points in order along the curve)

Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown curve

Output: the curve (the points in order along the curve)

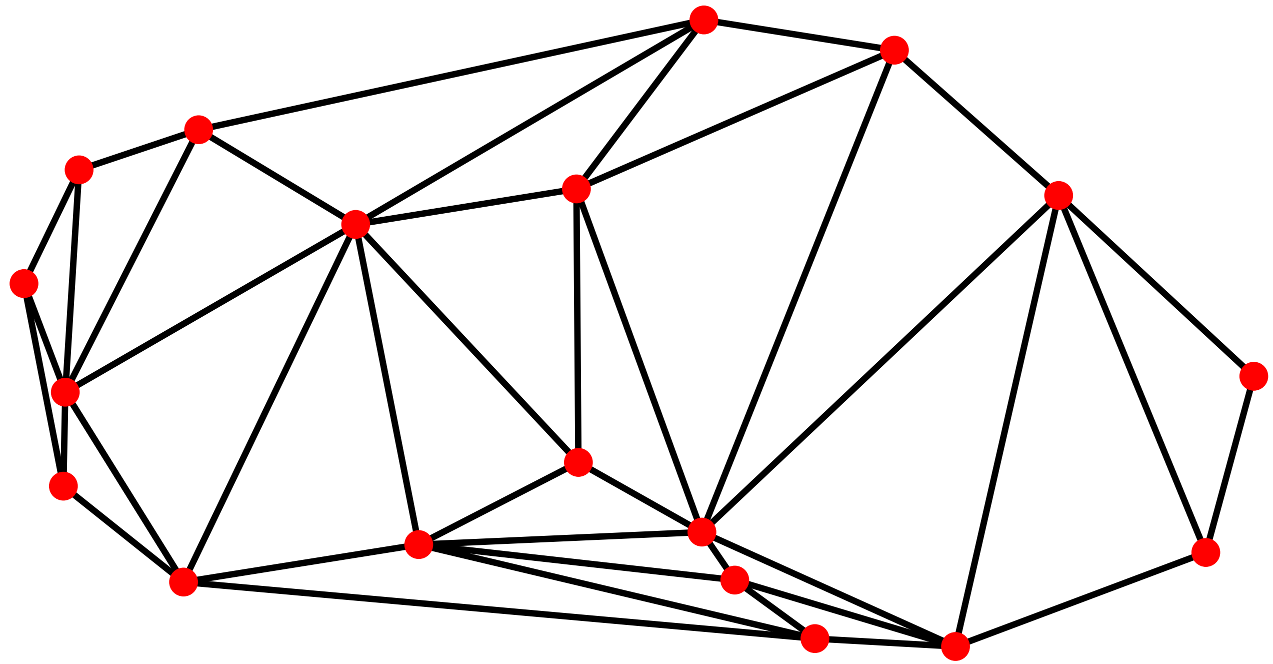


Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown curve

Output: the curve (the points in order along the curve)

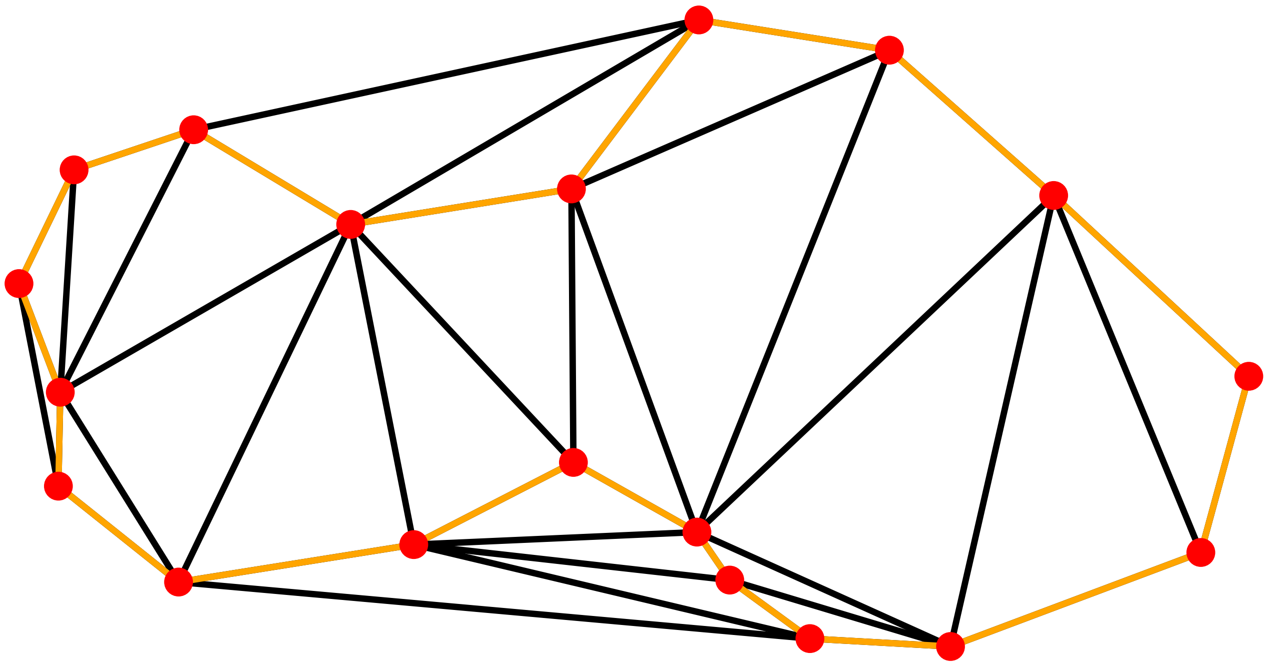


Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown curve

Output: the curve (the points in order along the curve)



If good sampling, ouput $\in$ Delaunay

Delaunay Triangulation:

Teaser reconstruction lecture

Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown surface

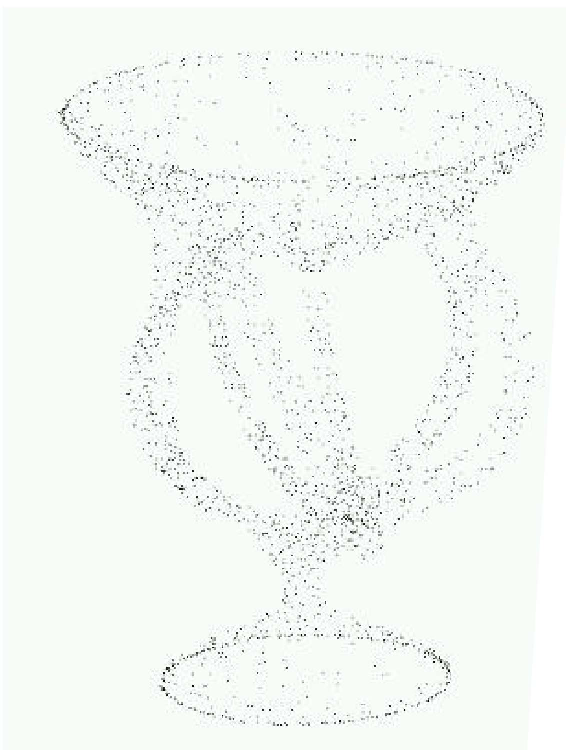
Output: the surface (a triangulation of the points approximating the surface)

Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown surface

Output: the surface (a triangulation of the points approximating the surface)

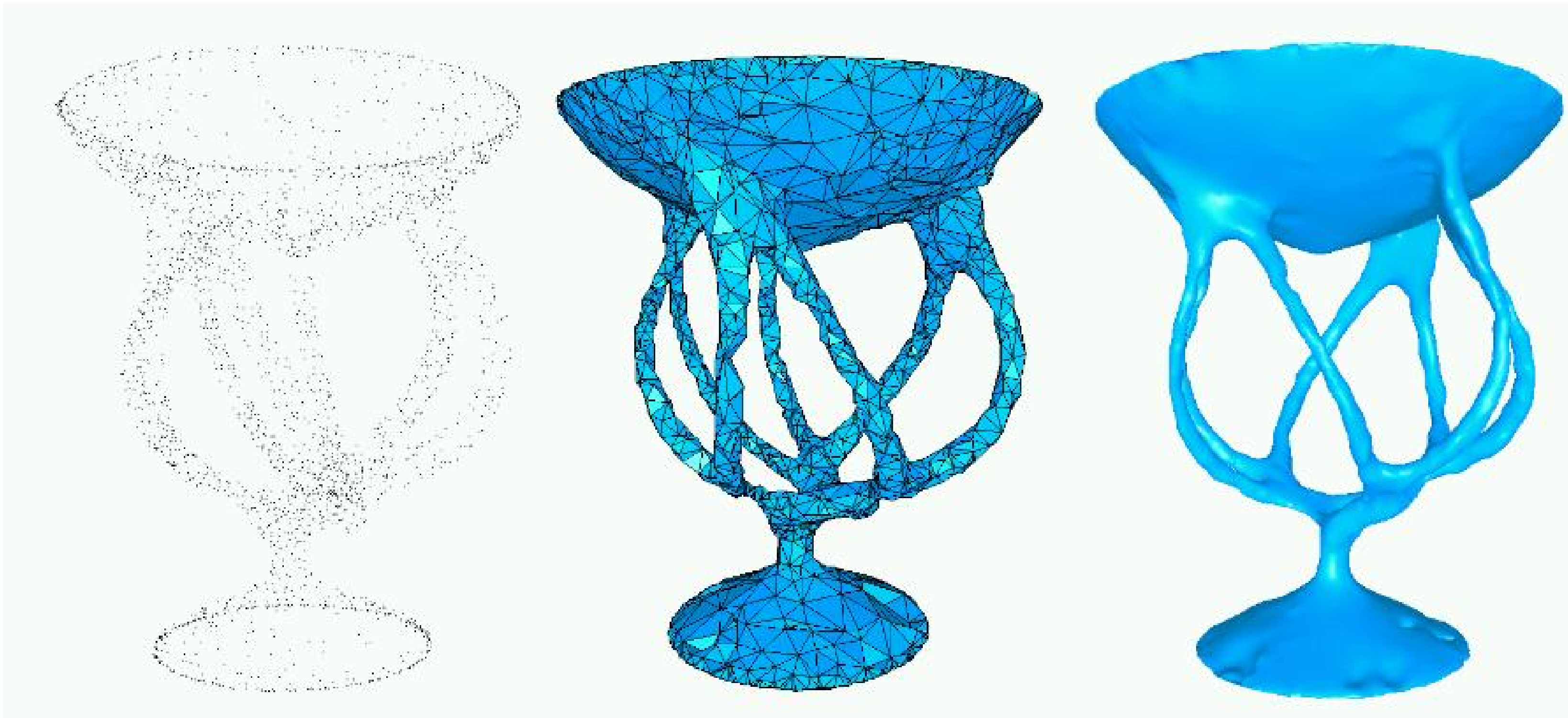


Delaunay Triangulation:

Teaser reconstruction lecture

Input: a set of points on an unknown surface

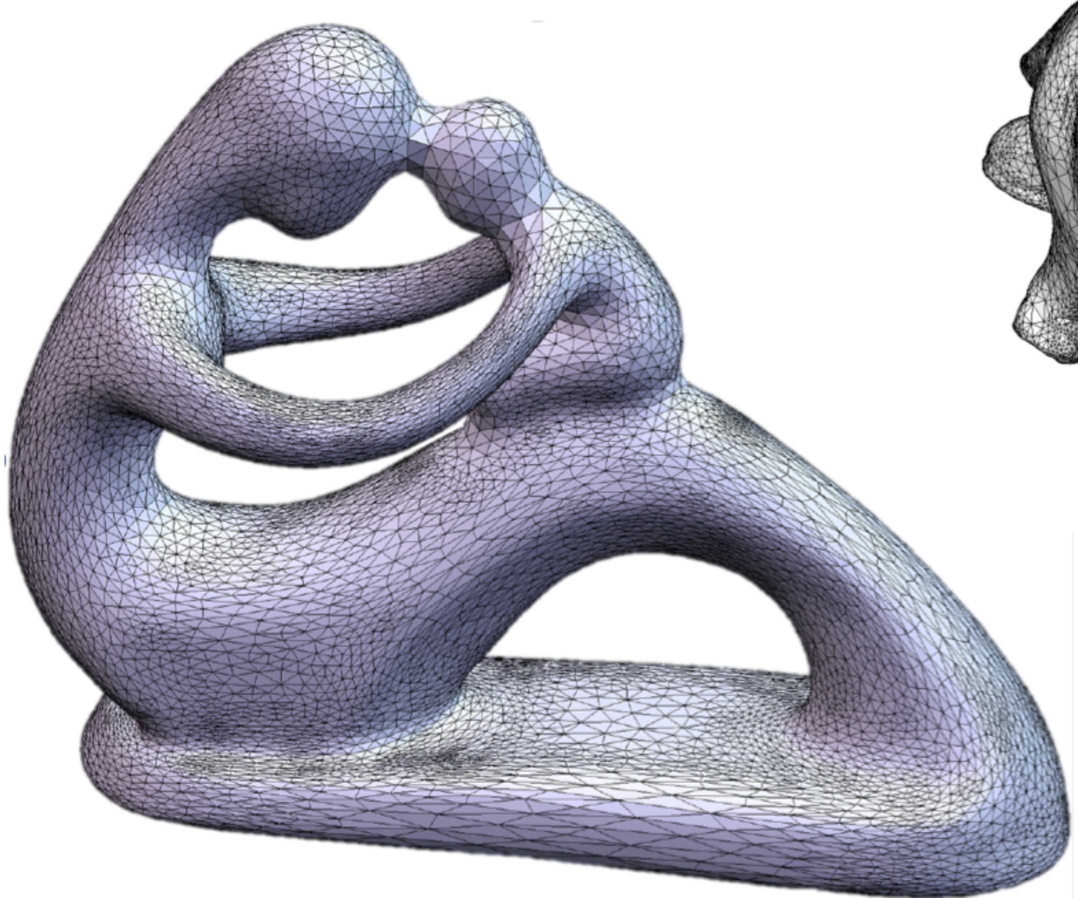
Output: the surface (a triangulation of the points approximating the surface)

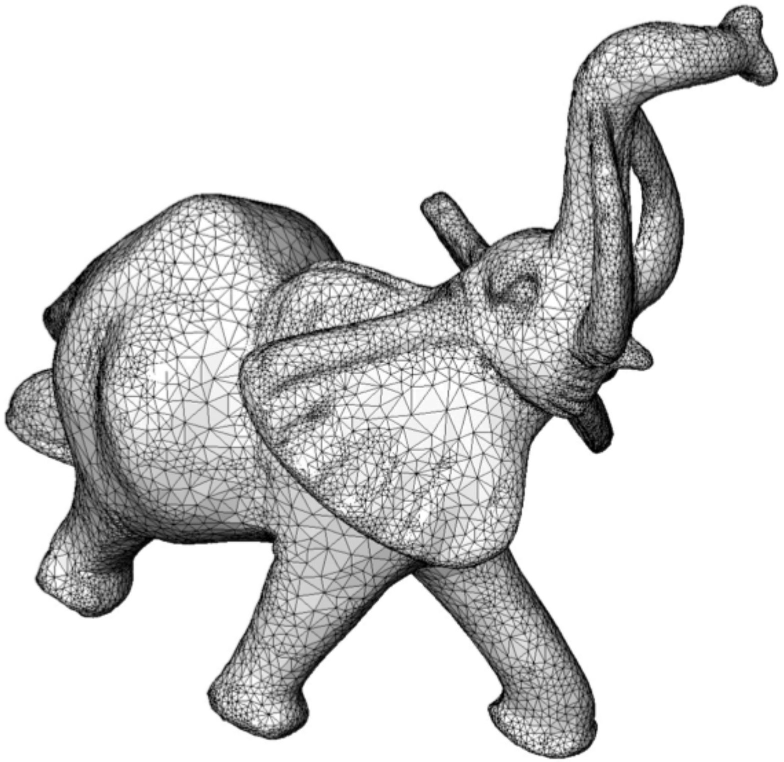


If good sampling, ouput $\in$ Delaunay

Delaunay Triangulation:

Teaser reconstruction lecture





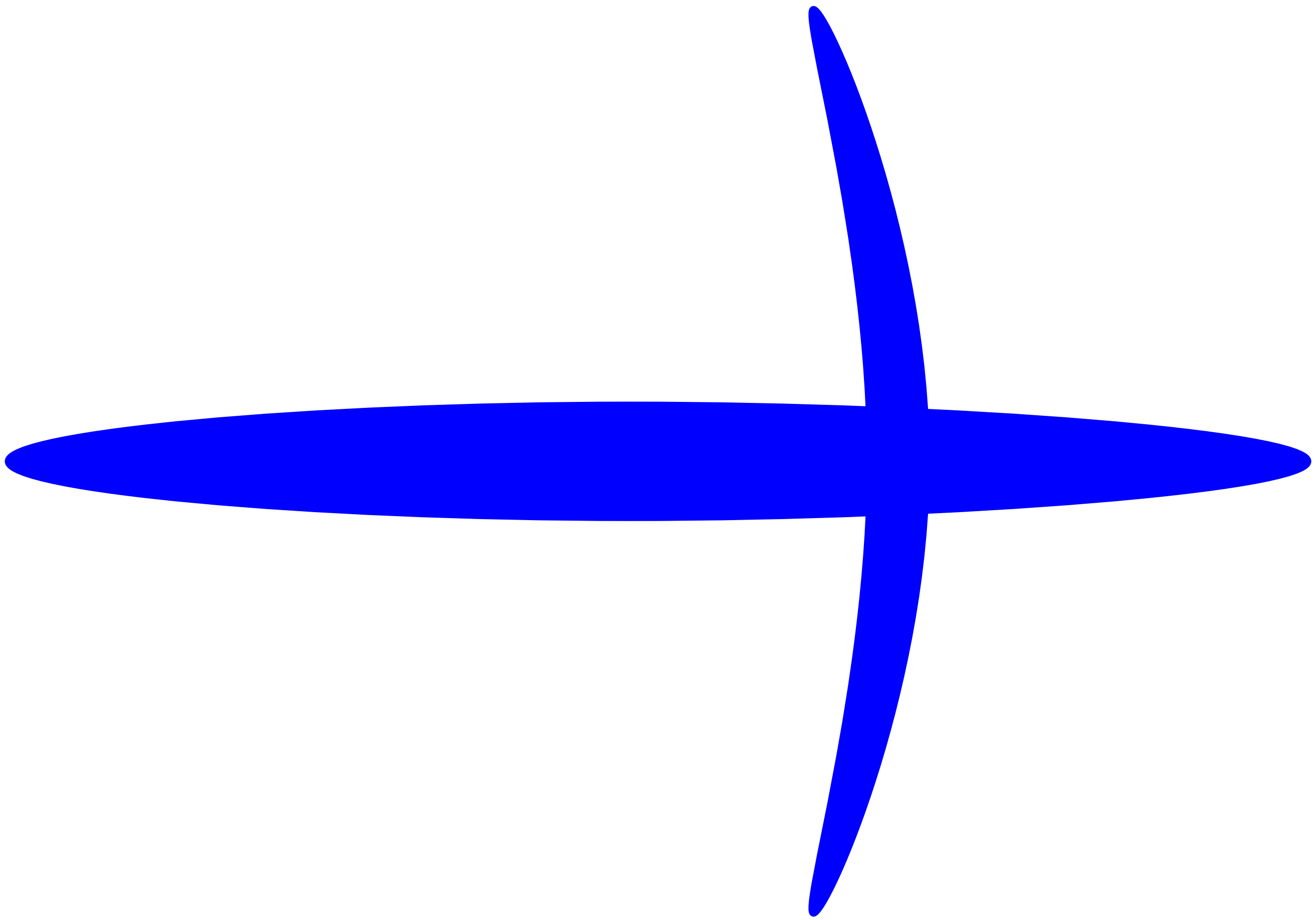
Delaunay Triangulation:

Teaser meshing lecture

Delaunay Triangulation:

Teaser meshing lecture

Shape



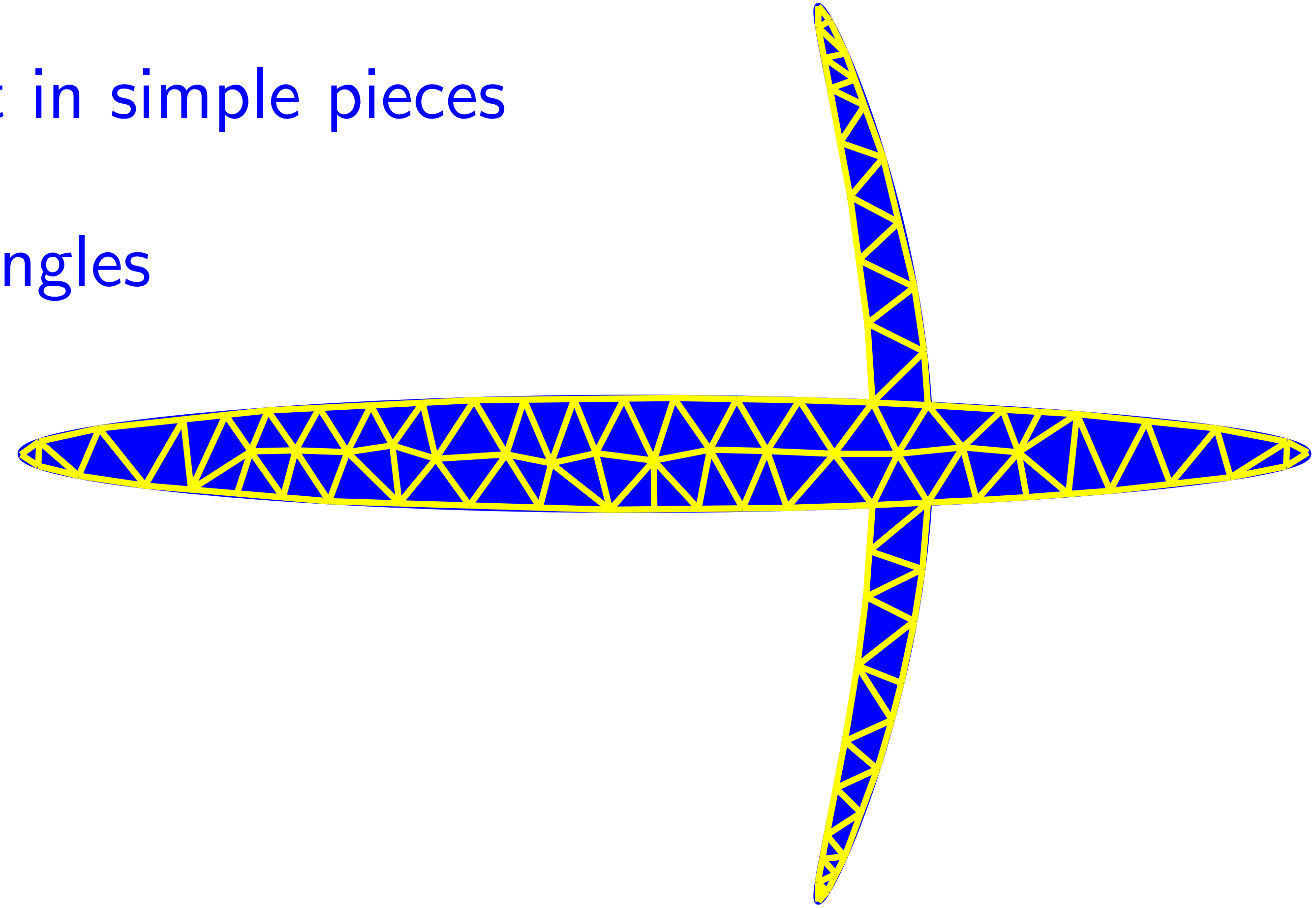
Delaunay Triangulation:

Teaser meshing lecture

Shape

Cut in simple pieces

triangles



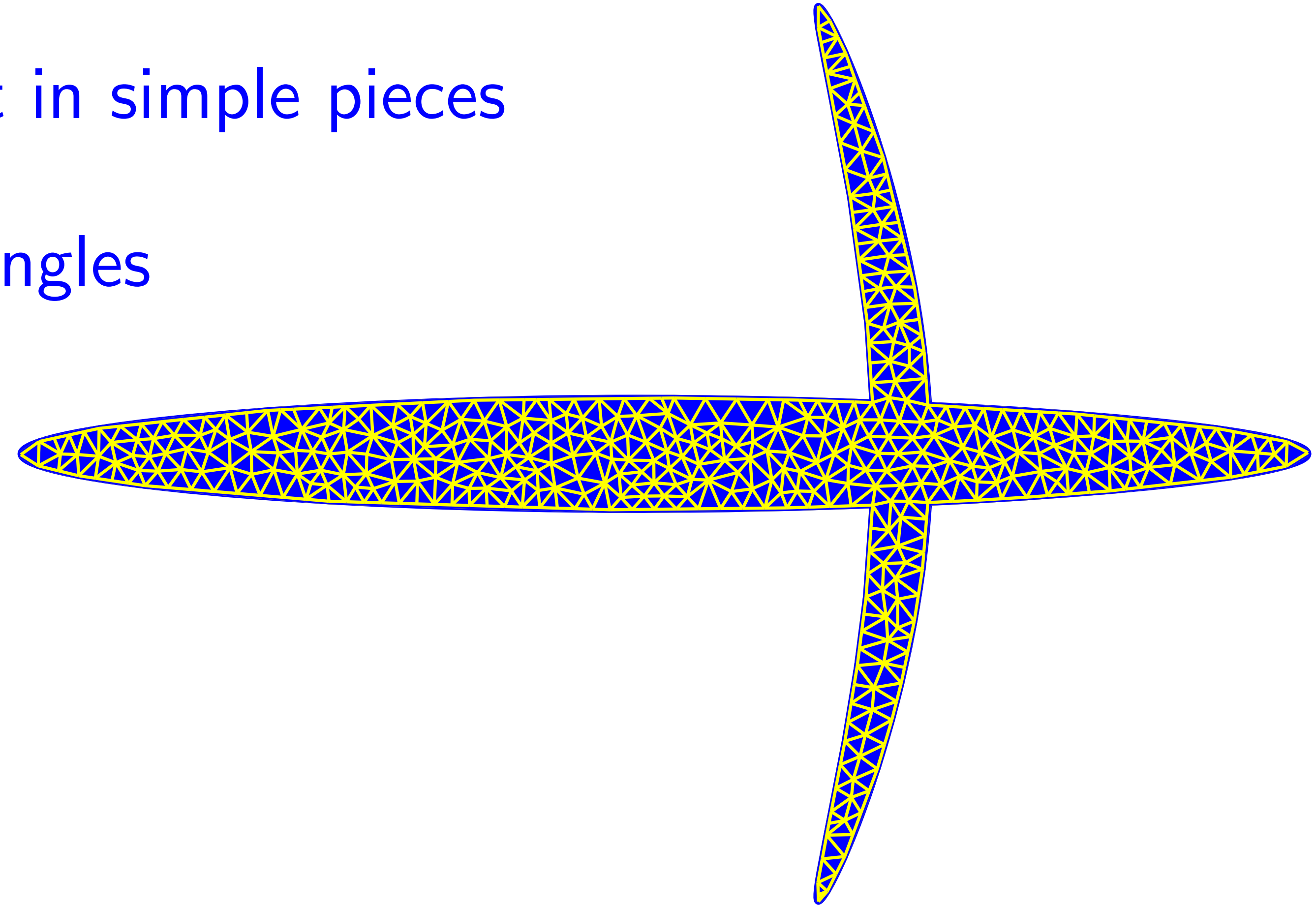
Delaunay Triangulation:

Teaser meshing lecture

Shape

Cut in simple pieces

triangles



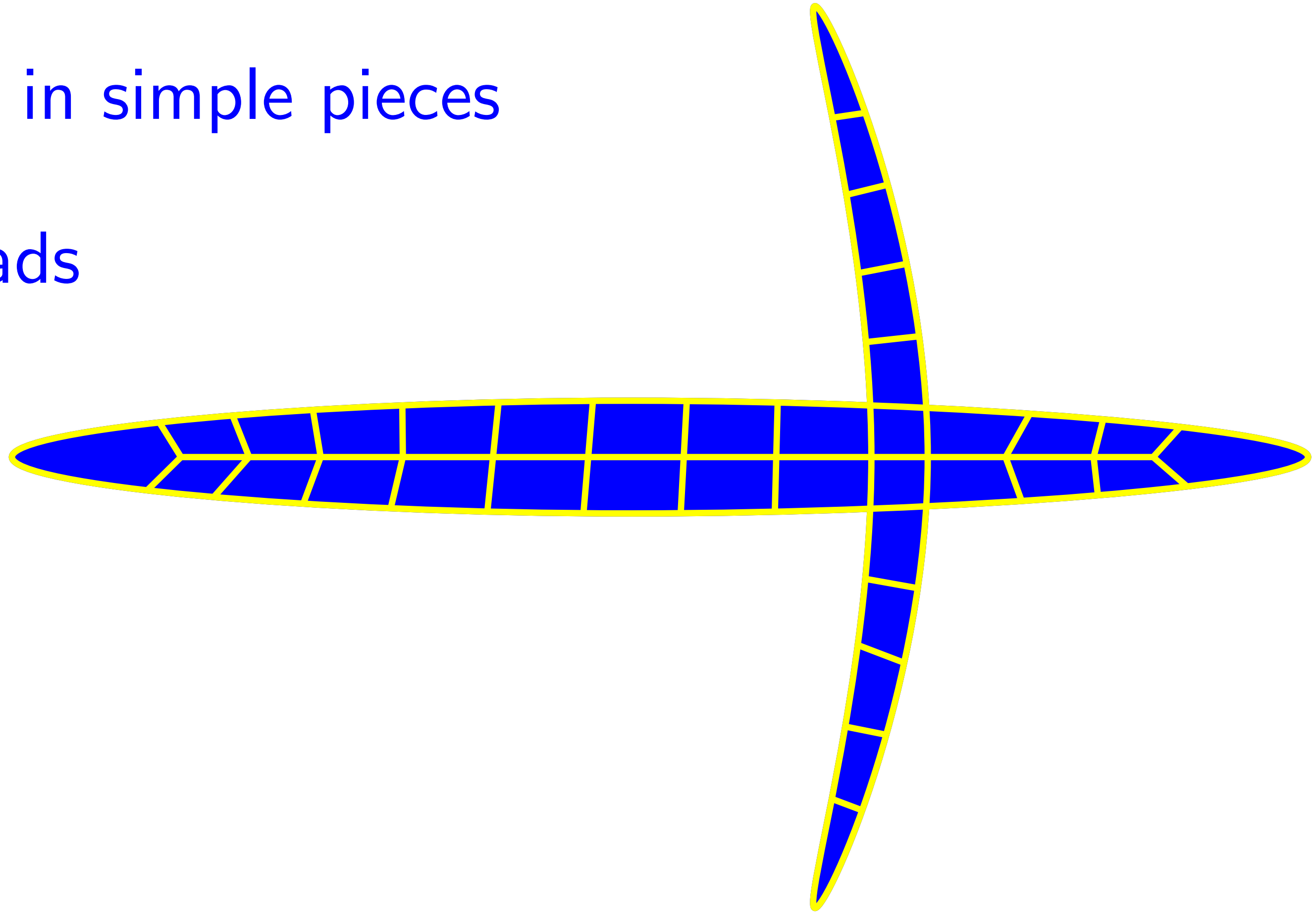
Delaunay Triangulation:

Teaser meshing lecture

Shape

Cut in simple pieces

Quads



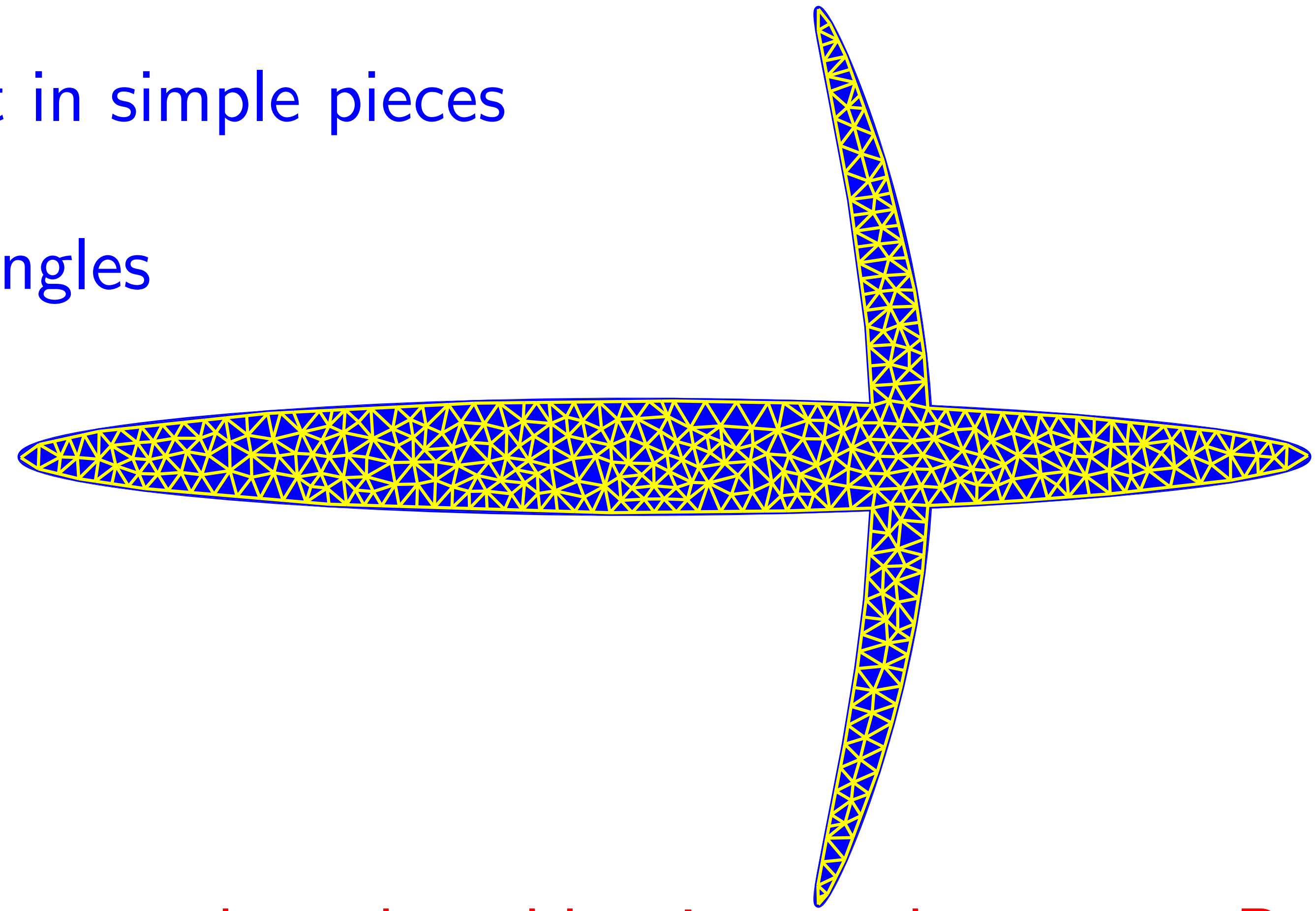
Delaunay Triangulation:

Teaser meshing lecture

Shape

Cut in simple pieces

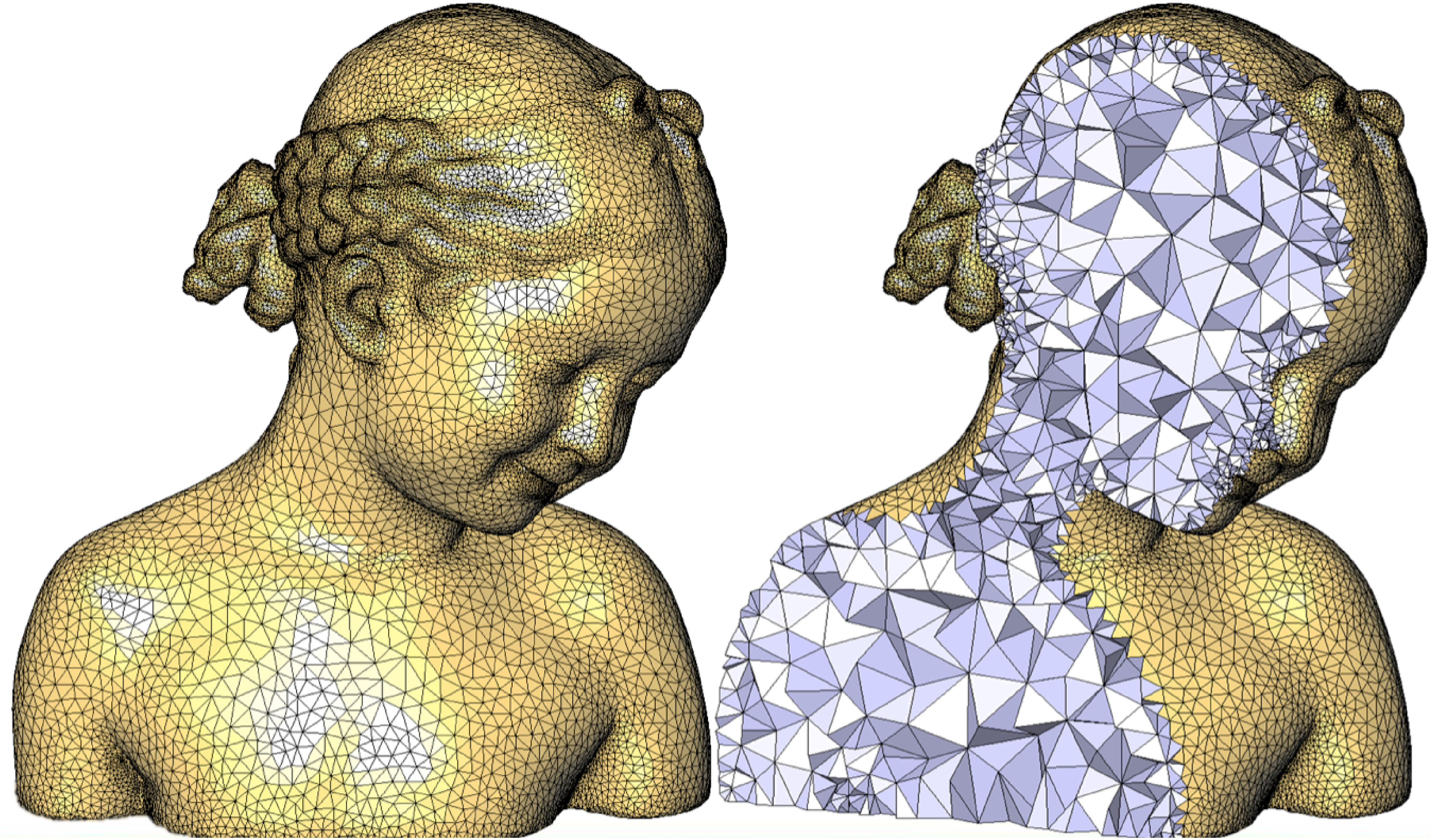
triangles



For unstructured mesh: add points and compute Delaunay

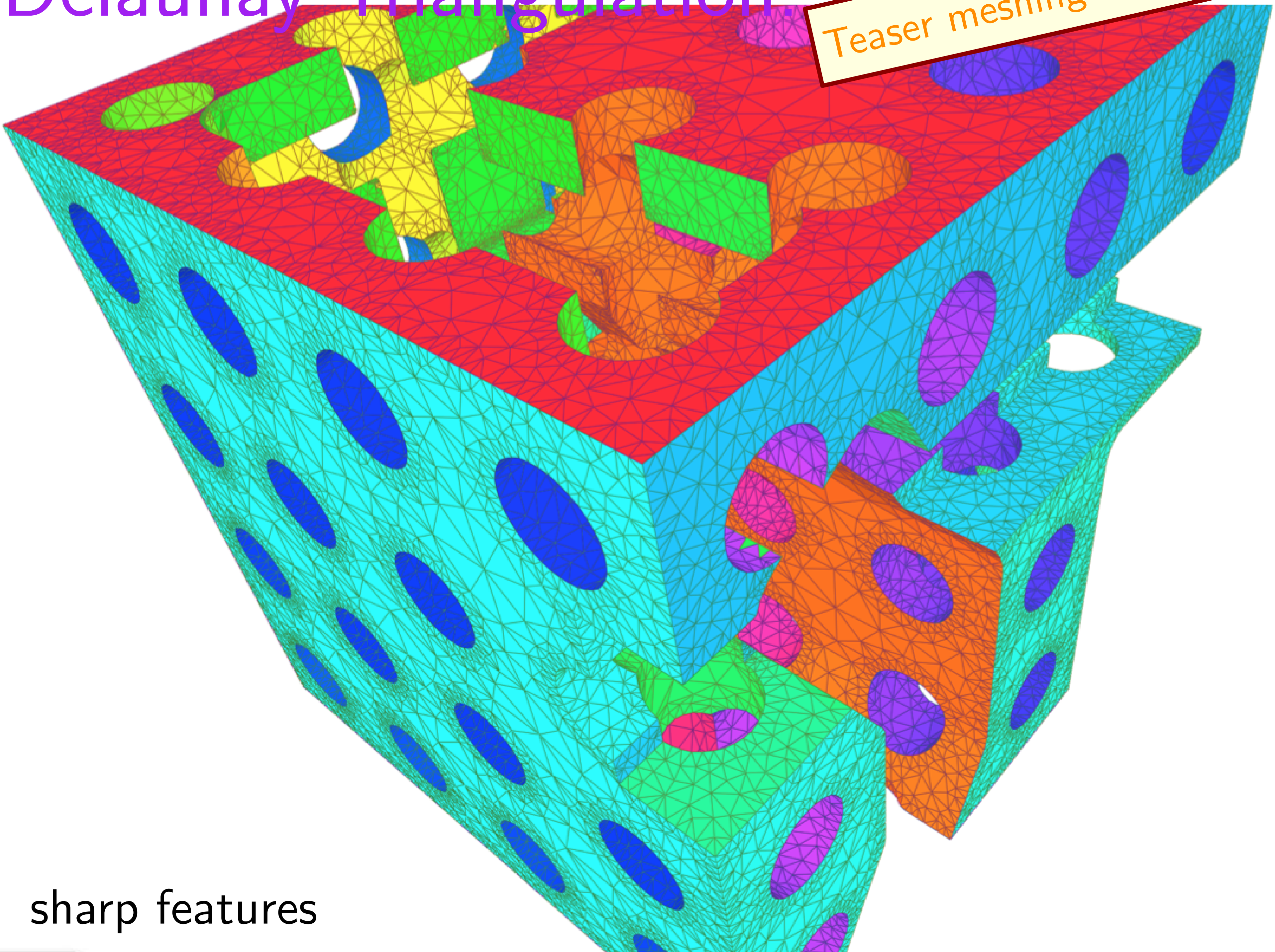
Delaunay Triangulation:

Teaser meshing lecture



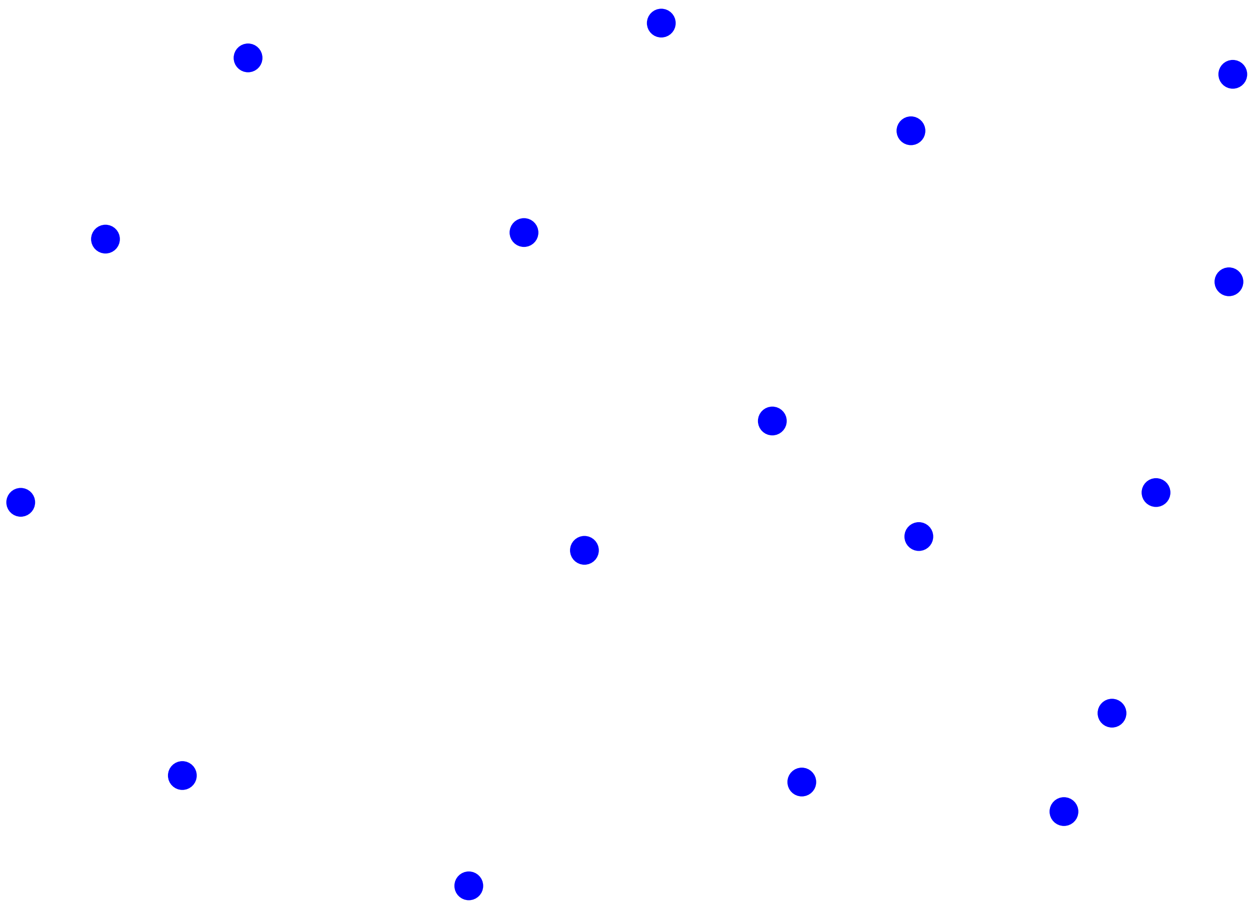
Delaunay Triangulation:

Teaser meshing lecture



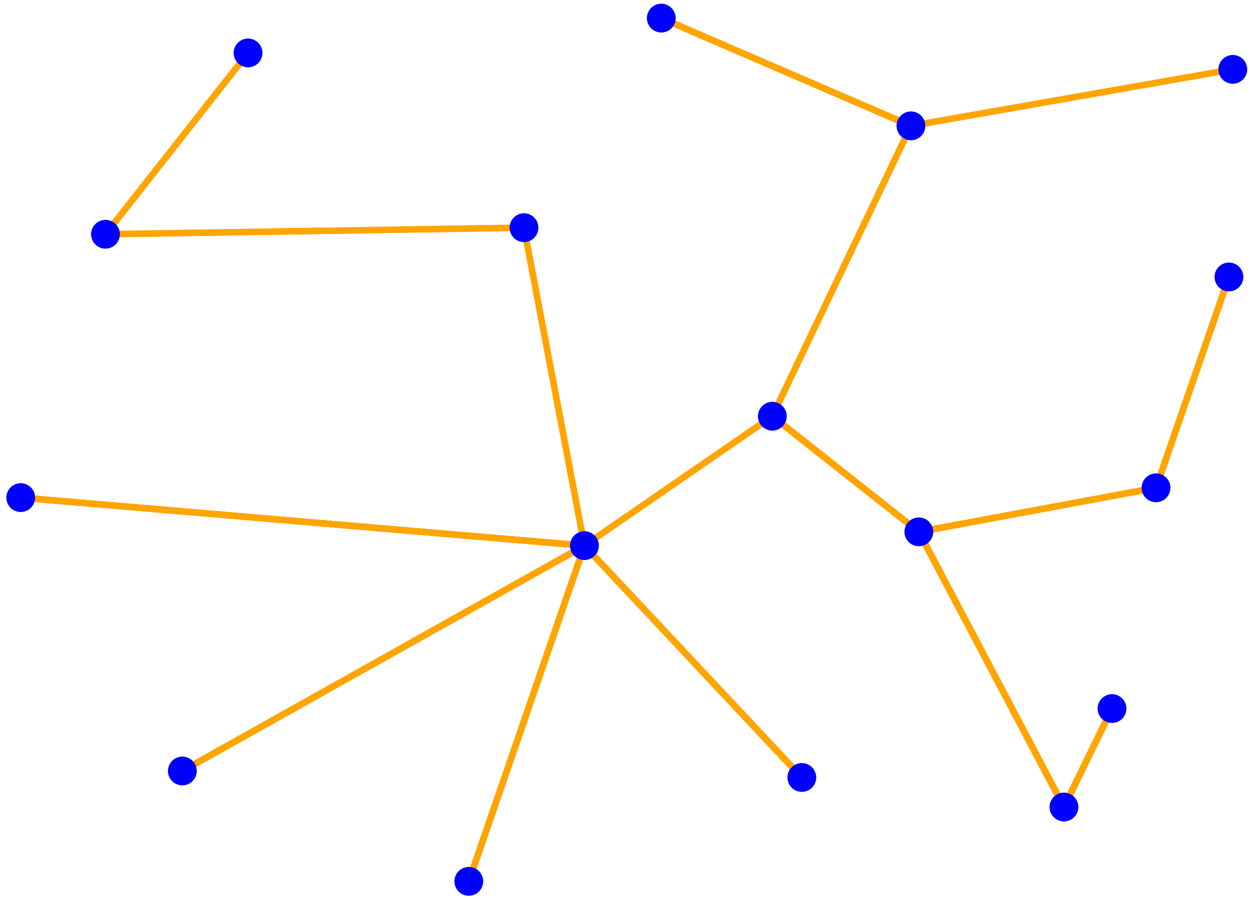
sharp features

Delaunay Triangulation: EMST



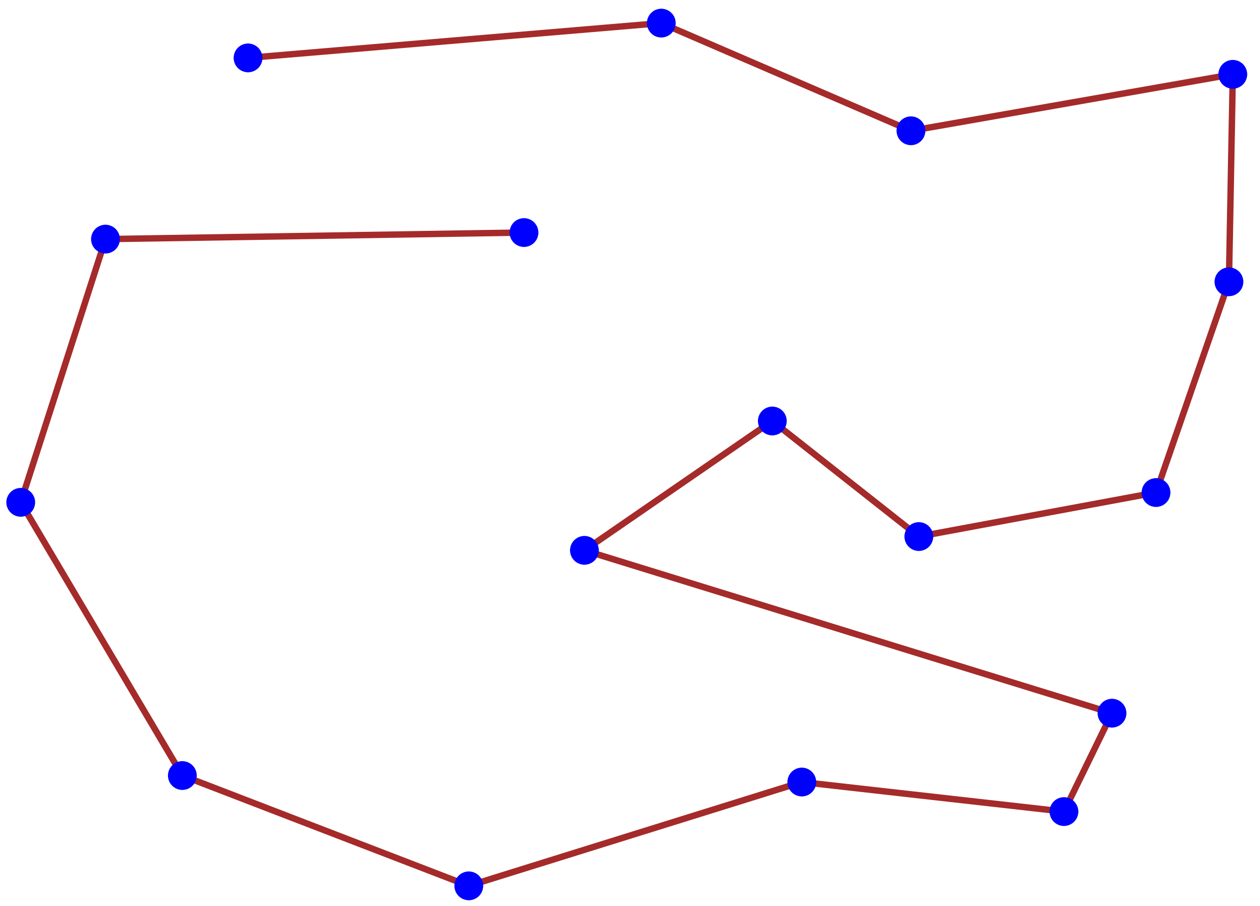
Delaunay Triangulation: EMST

A spanning tree



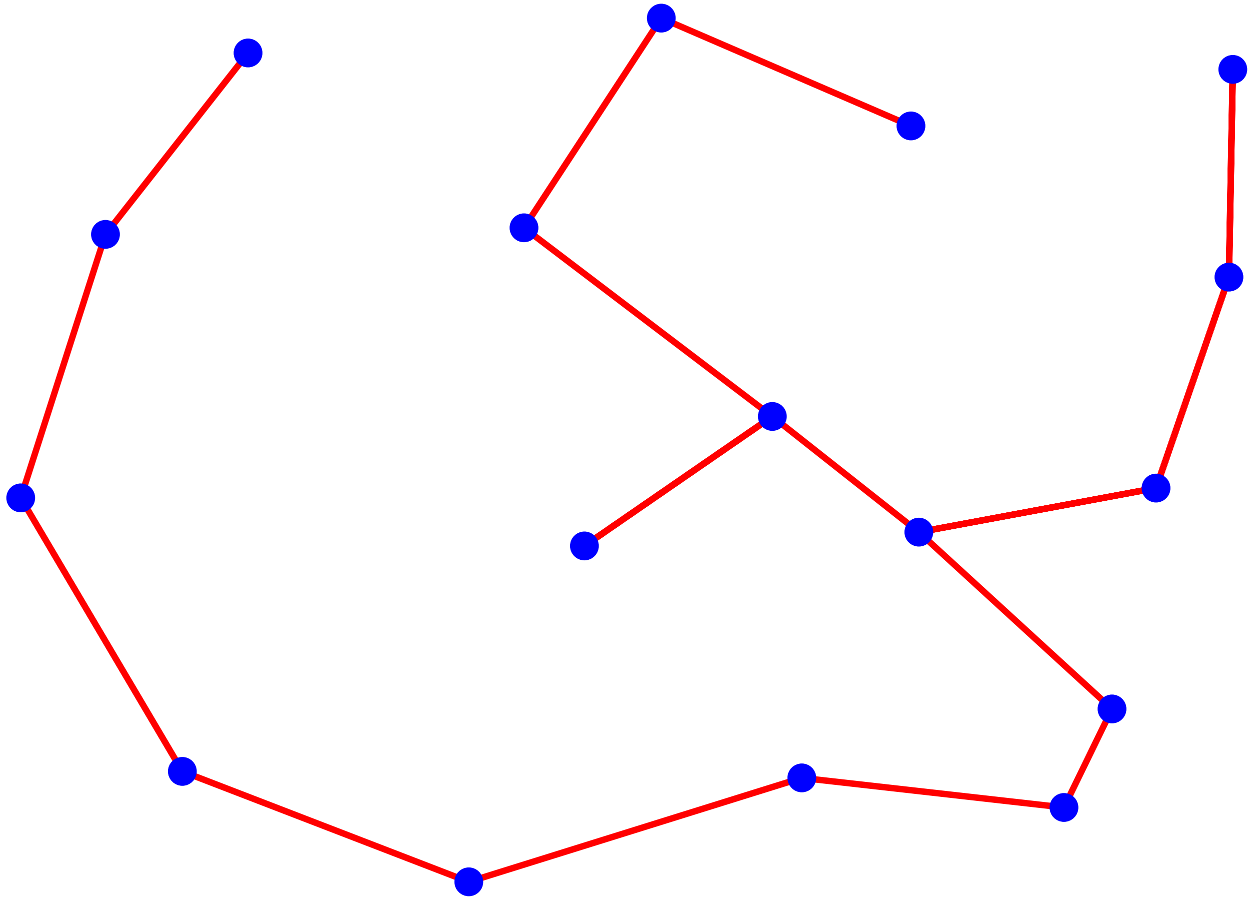
Delaunay Triangulation: EMST

Another spanning tree



Delaunay Triangulation: EMST

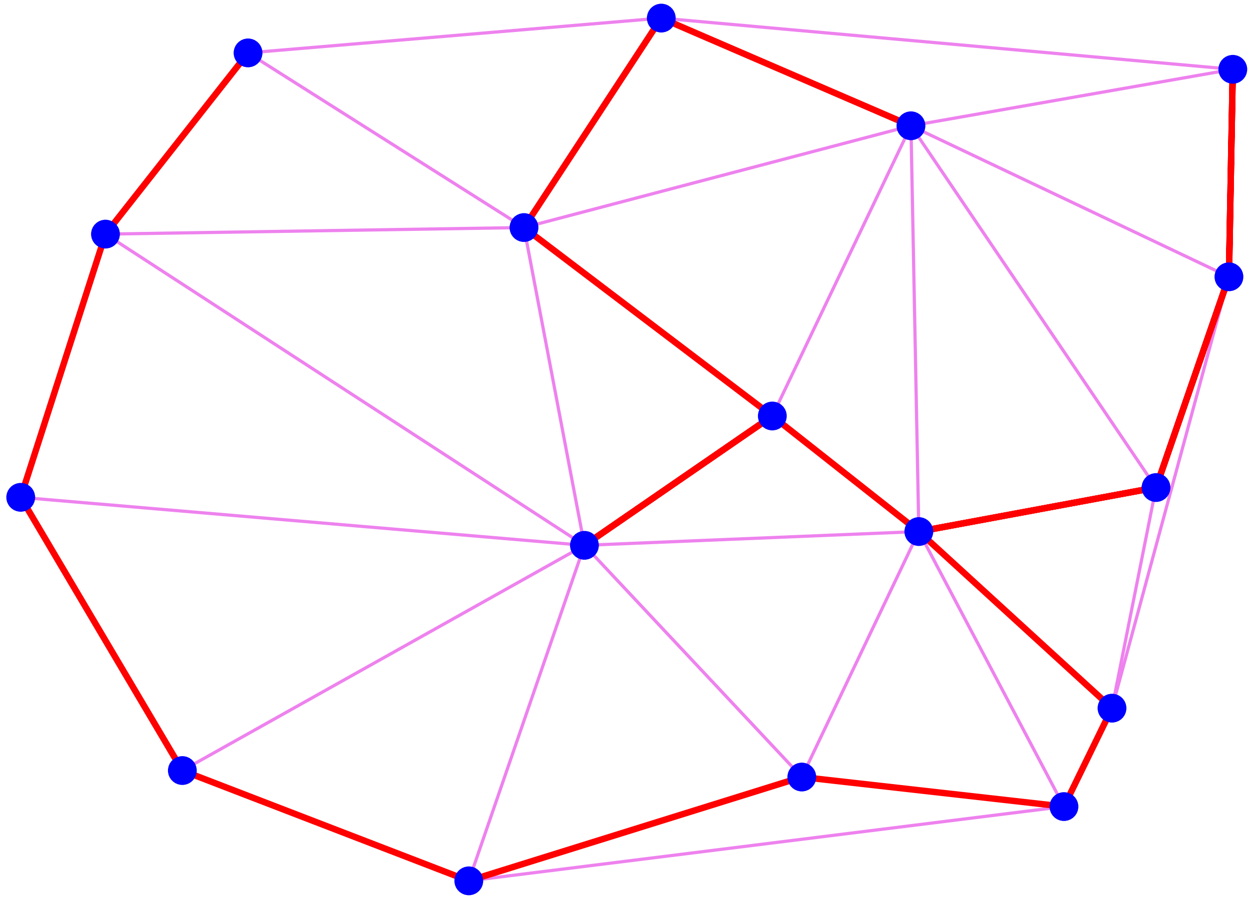
The Euclidean Minimum-length Spanning Tree



Delaunay Triangulation: EMST

The Euclidean Minimum-length Spanning Tree

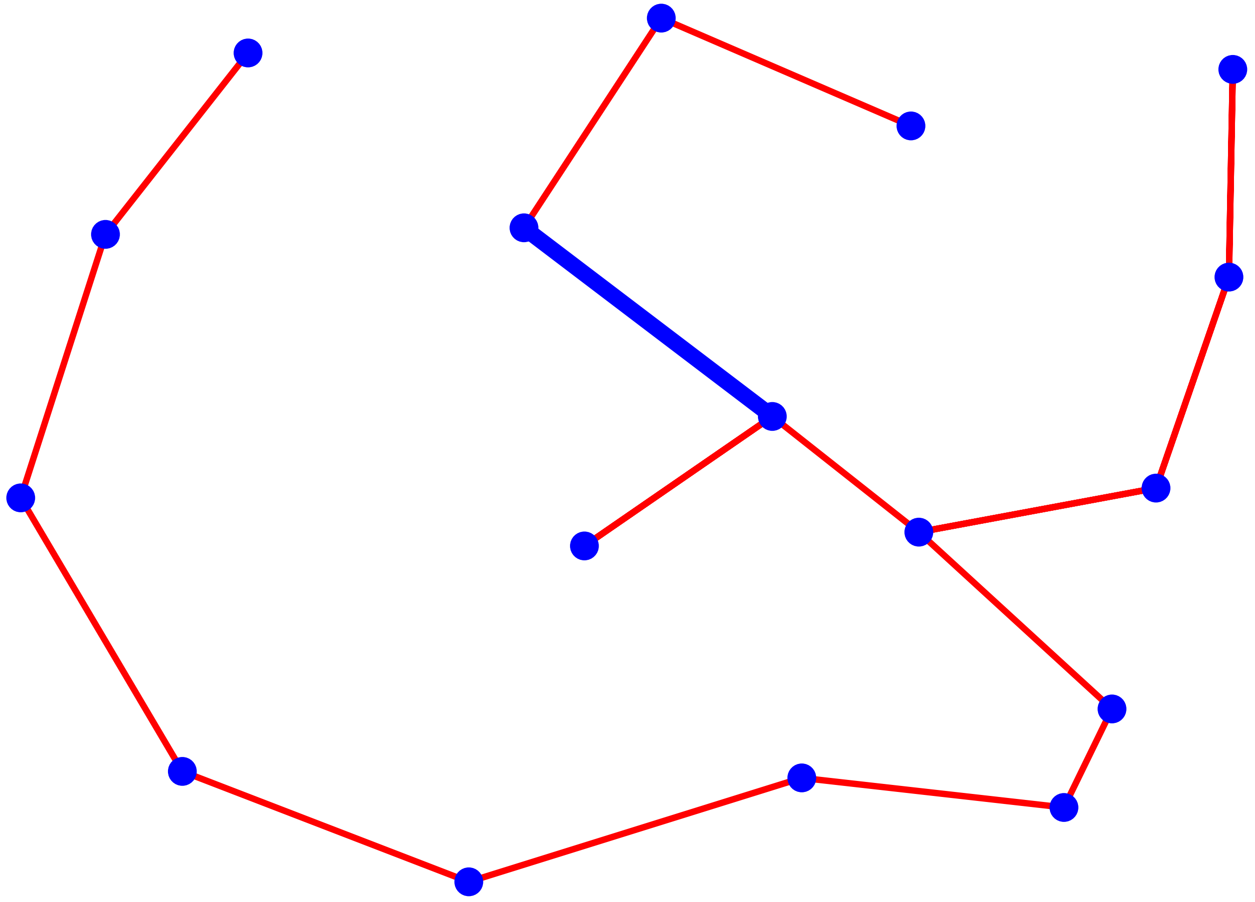
is included in Delaunay



Delaunay Triangulation: EMST

The Euclidean Minimum-length Spanning Tree
is included in Delaunay

Proof:

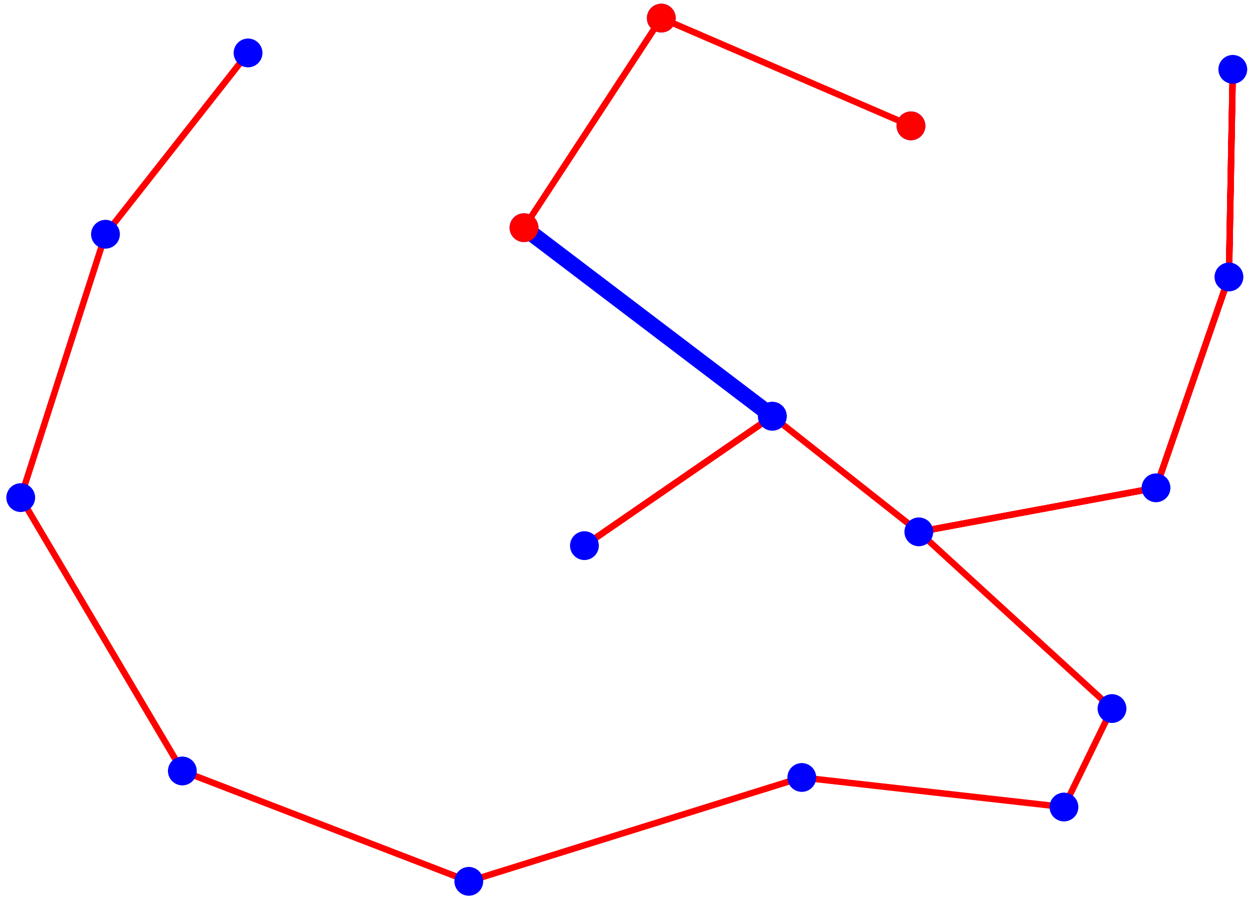


Choose an edge of EMST

Delaunay Triangulation: EMST

The Euclidean Minimum-length Spanning Tree

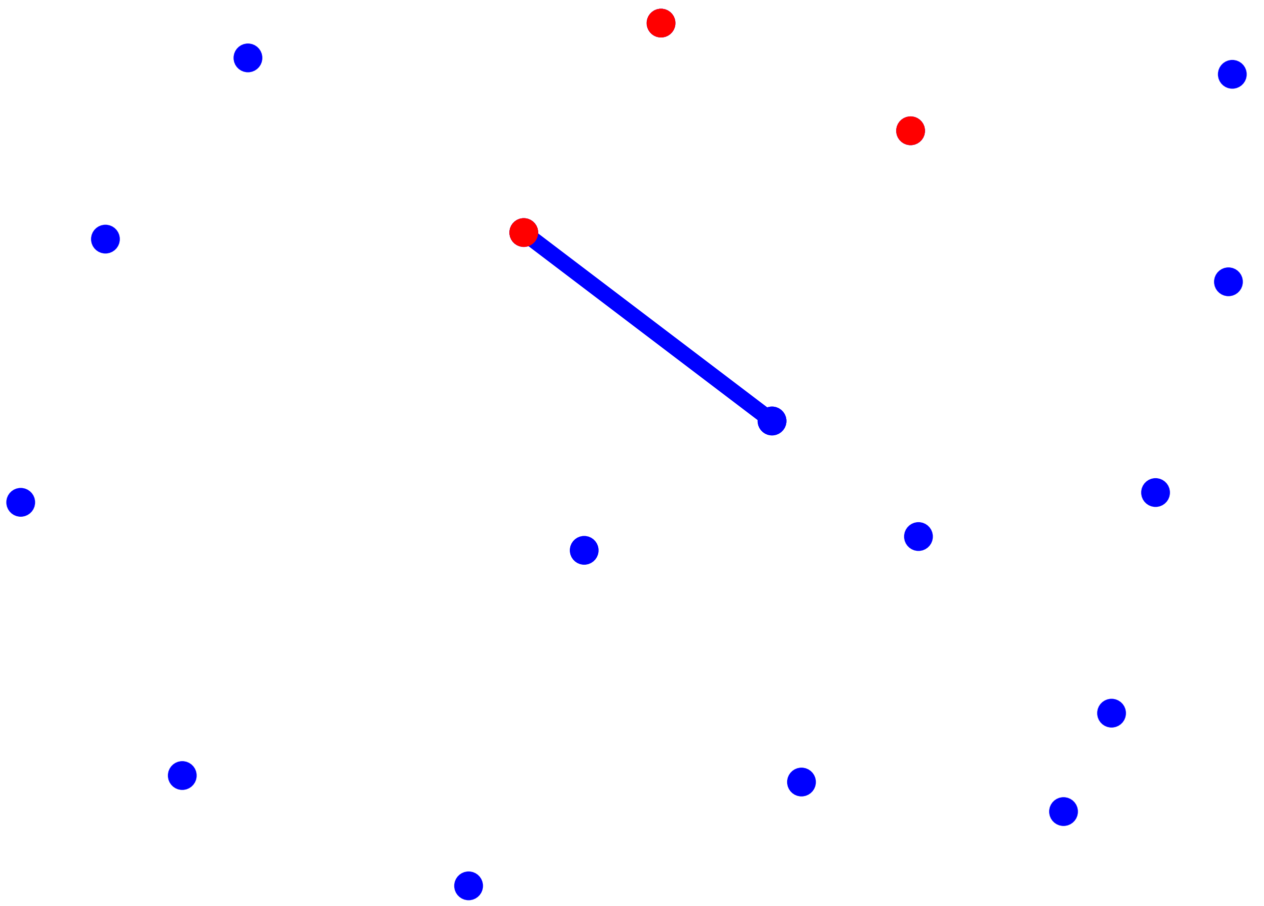
Proof:



Split points

Delaunay Triangulation: EMST

Proof:

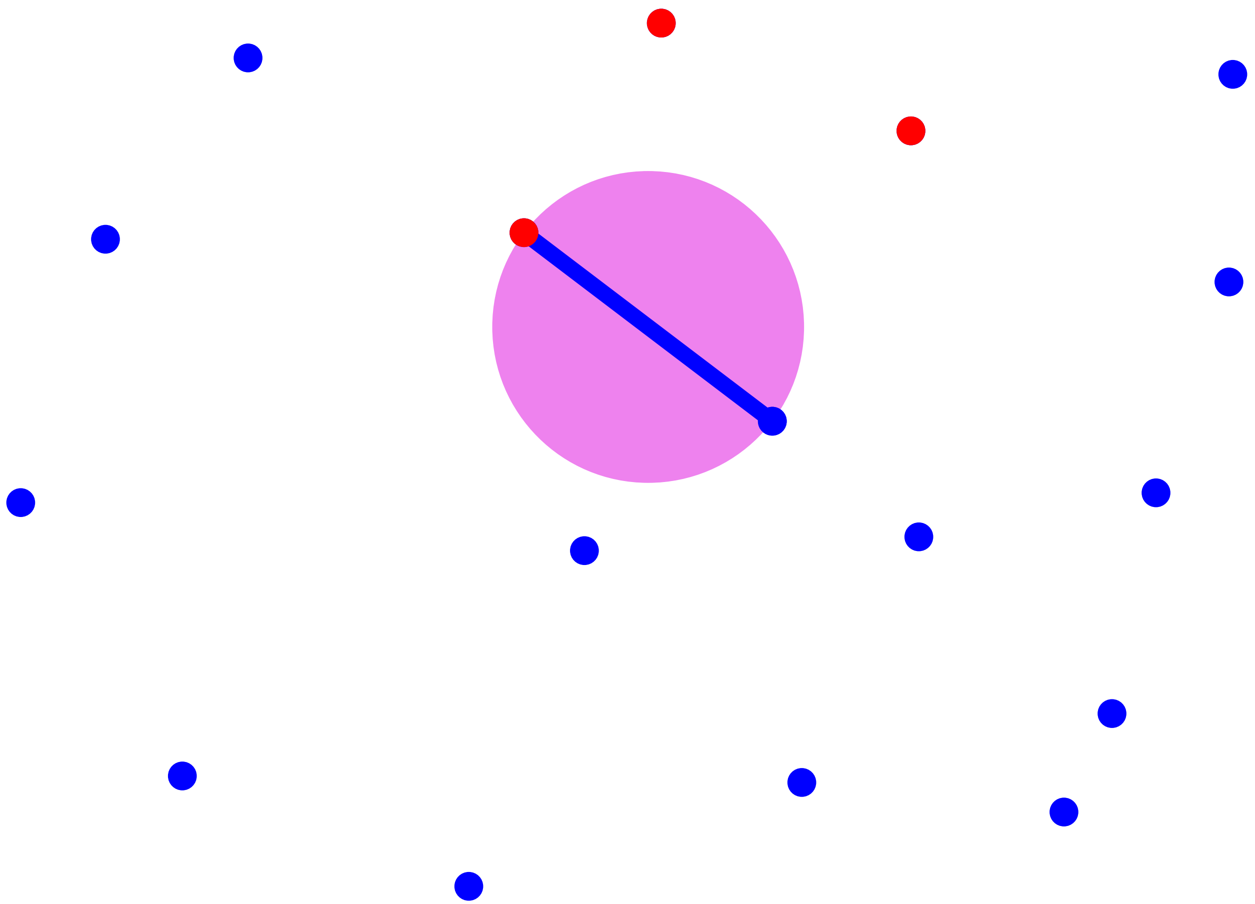


Split points

Delaunay Triangulation: EMST

Is diametral circle empty ?

Proof:

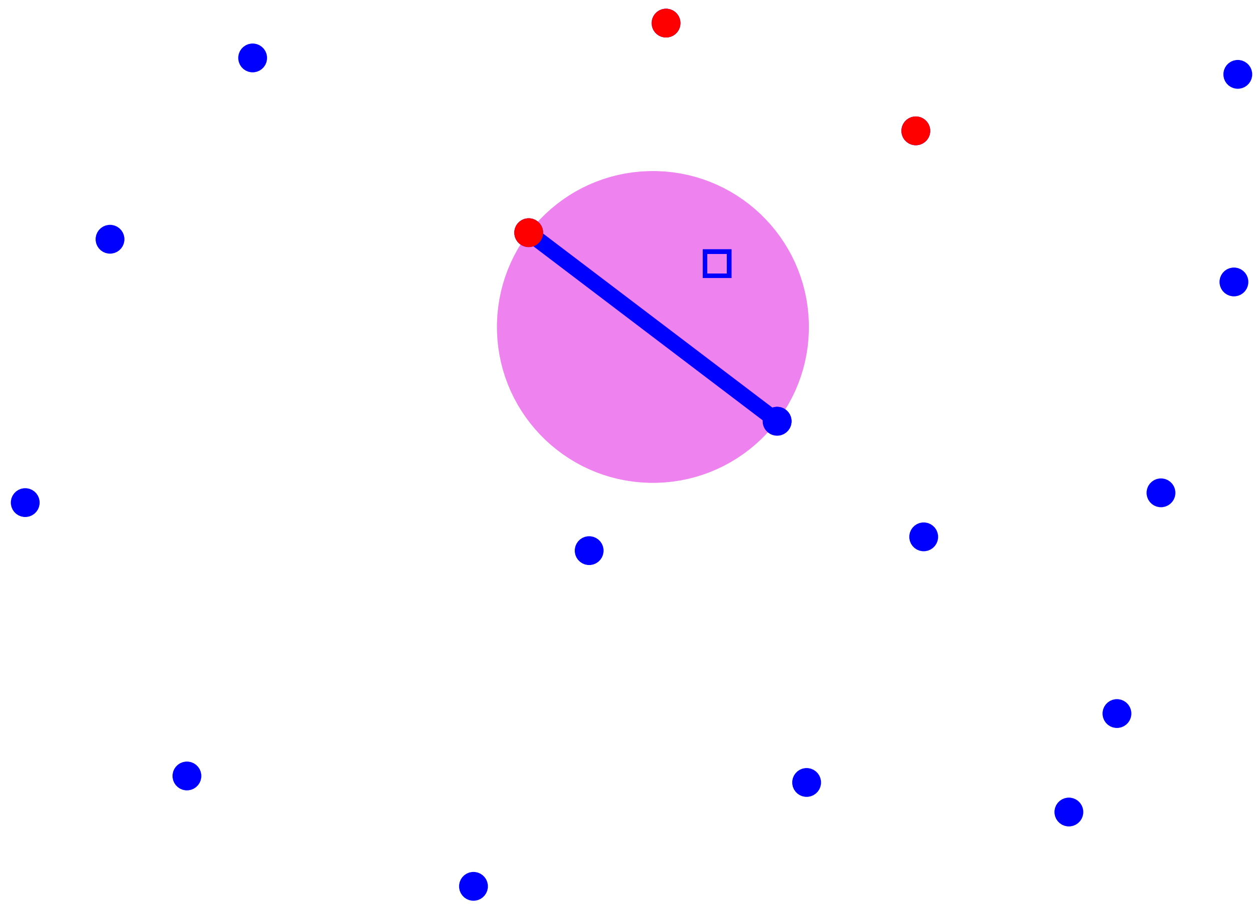


Split points

Delaunay Triangulation: EMST

Is diametral circle empty ?   assume ∃ blue point inside

Proof:



Split points

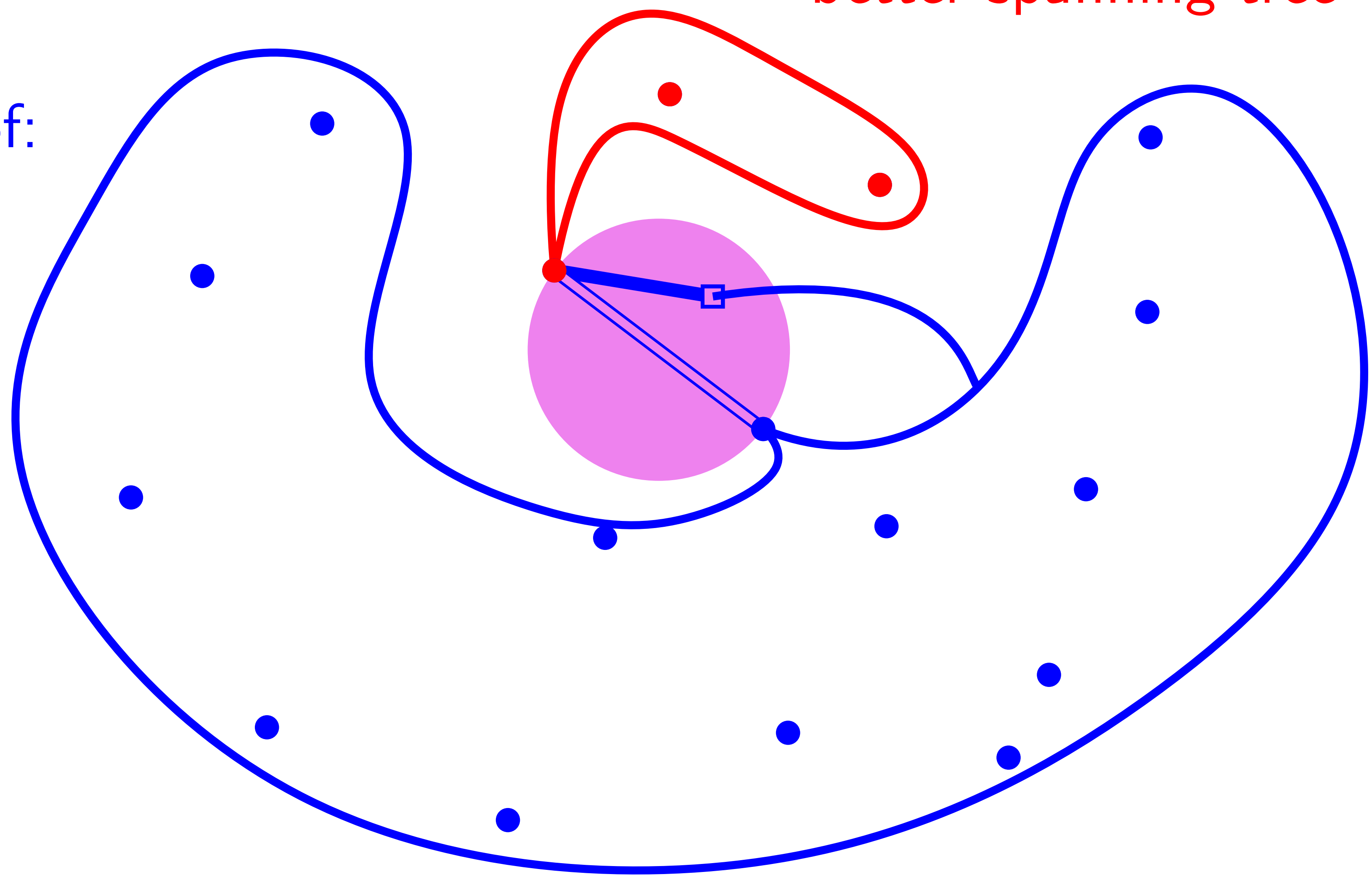
Delaunay Triangulation: EMST

Is diametral circle empty ?

assume ∃ blue point inside

better spanning tree

Proof:



Split points

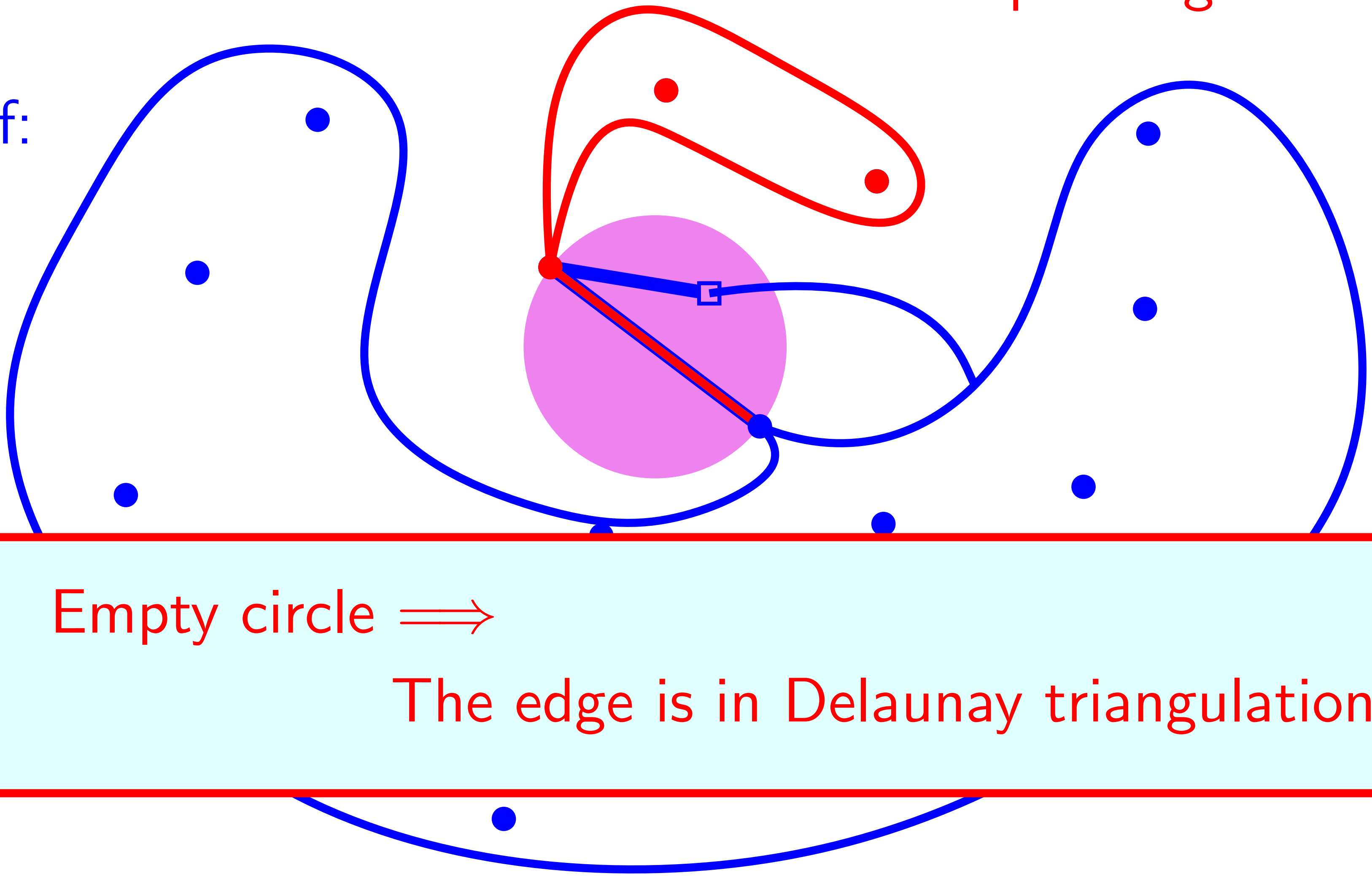
Delaunay Triangulation: EMST

Is diametral circle empty ?

assume ∃ blue point inside

better spanning tree

Proof:



Empty circle $\Longrightarrow$

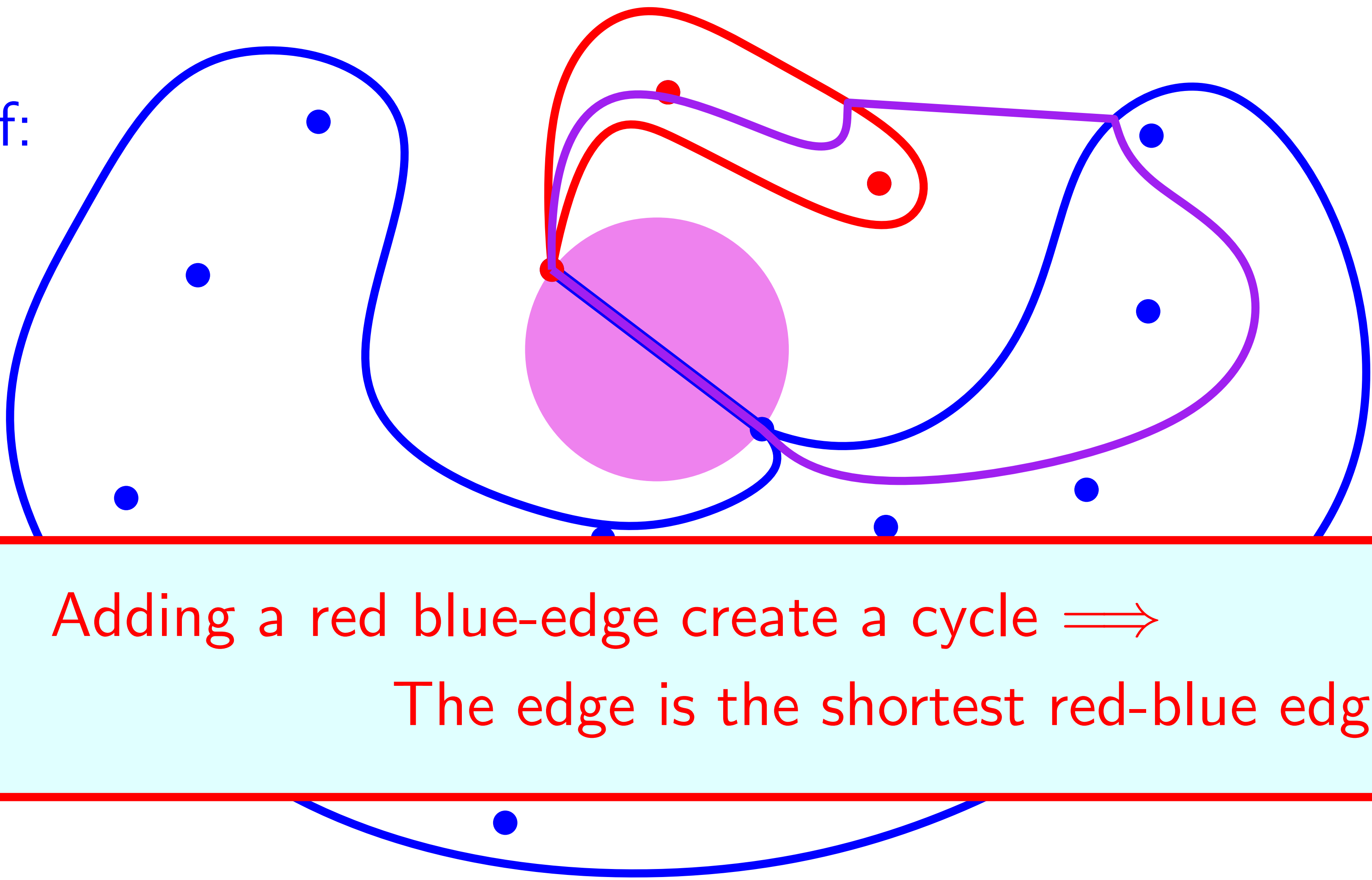
The edge is in Delaunay triangulation

Split points

Delaunay Triangulation: EMST

Is diametral circle empty ?

Proof:

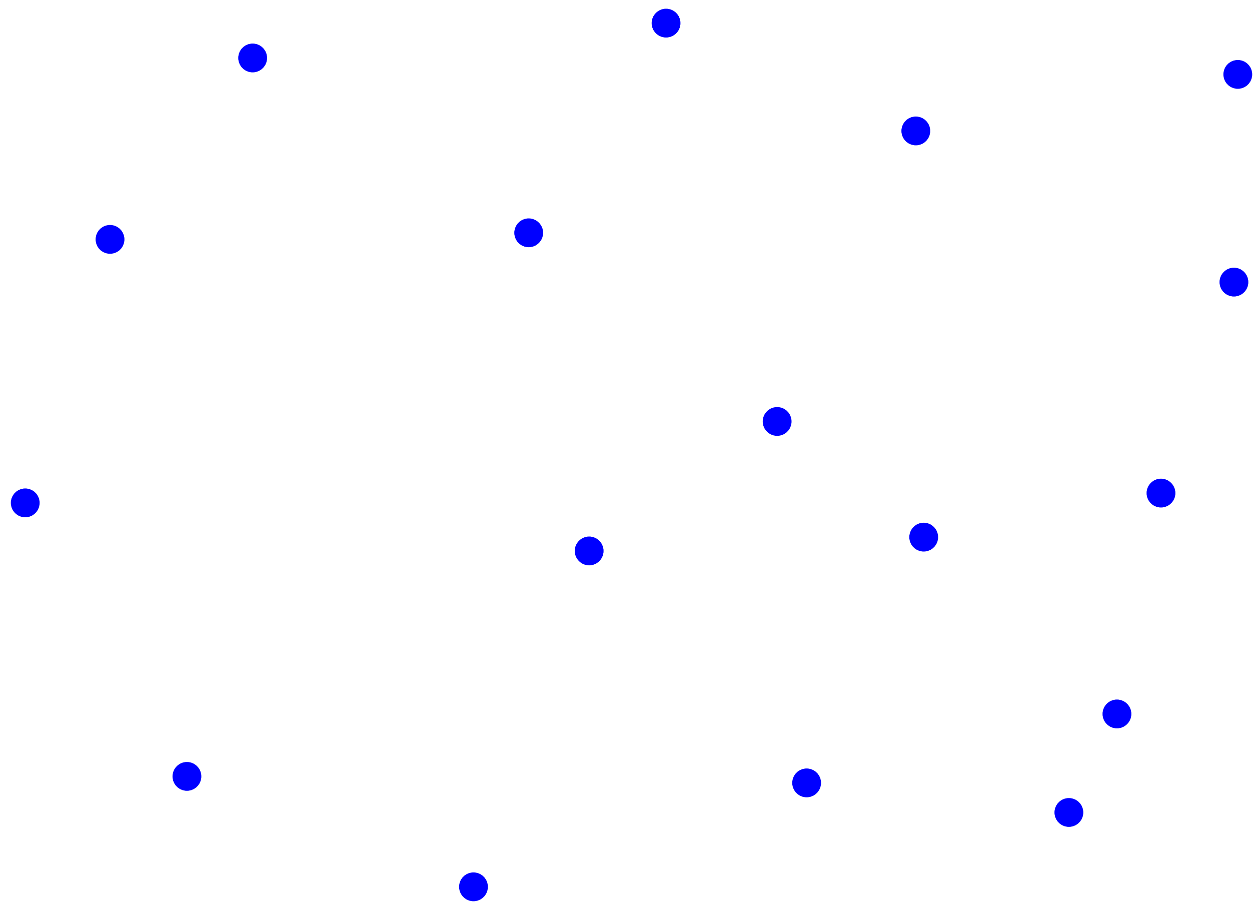


Adding a red blue-edge create a cycle $\Longrightarrow$

The edge is the shortest red-blue edge

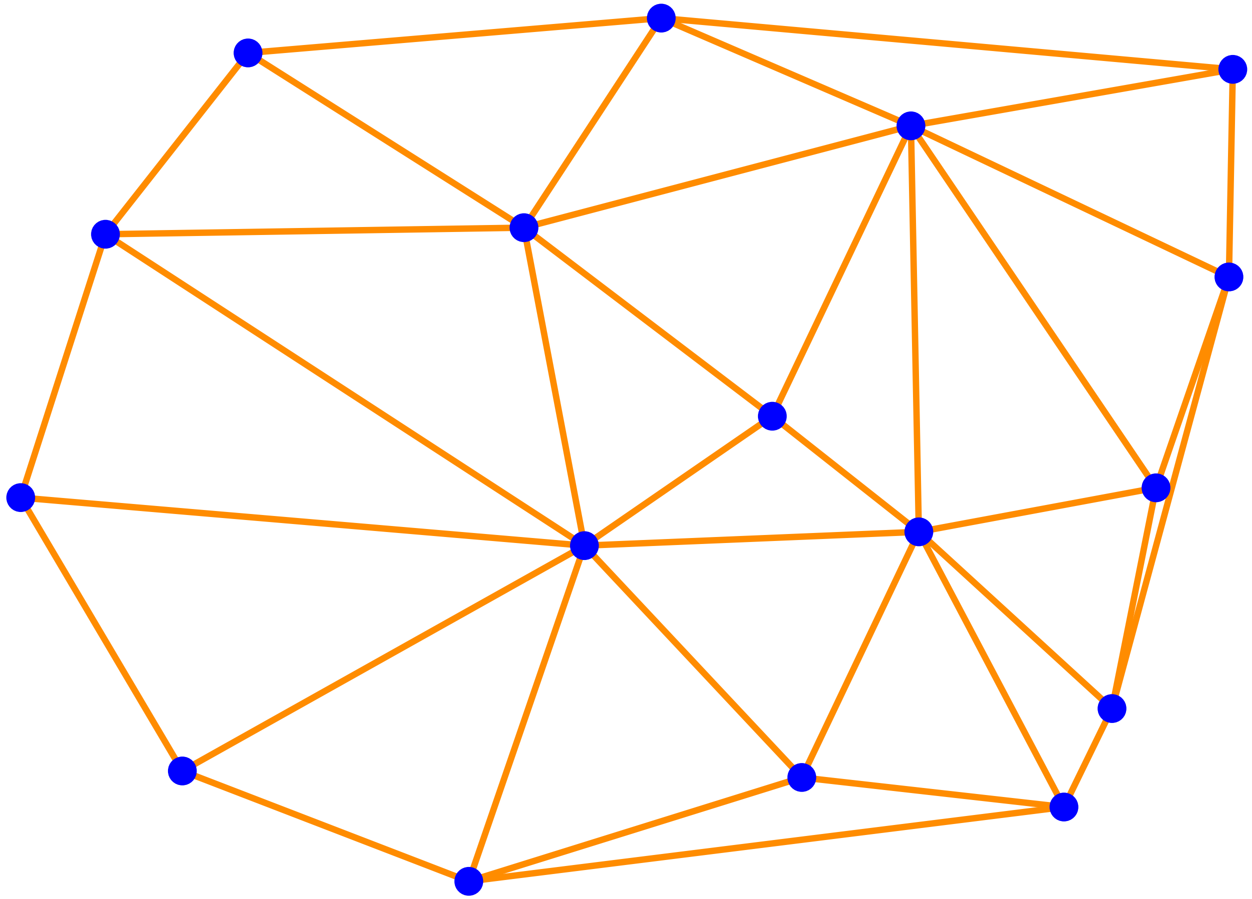
Delaunay Triangulation: EMST

Algorithm



Delaunay Triangulation: EMST

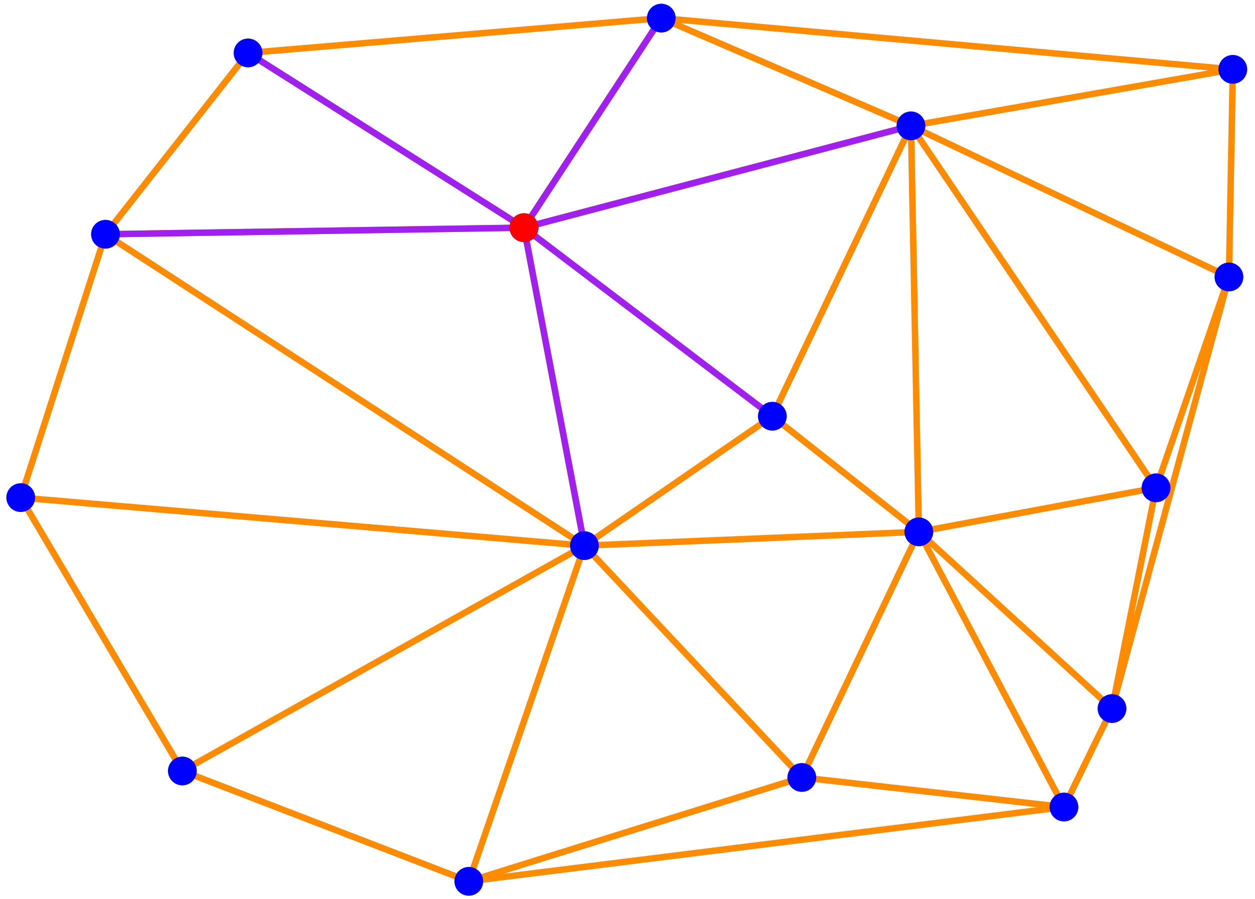
Algorithm



Delaunay Triangulation: EMST

Algorithm

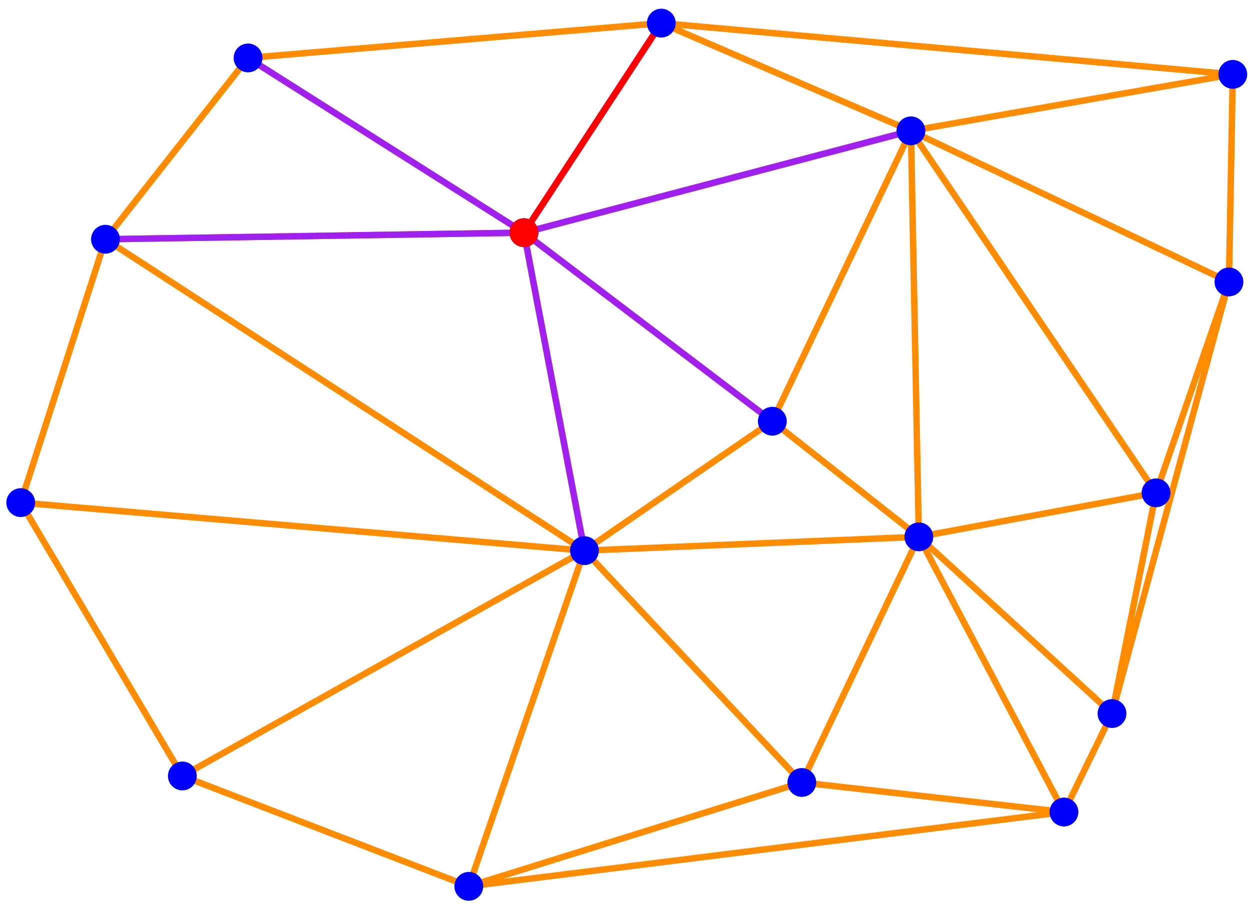
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

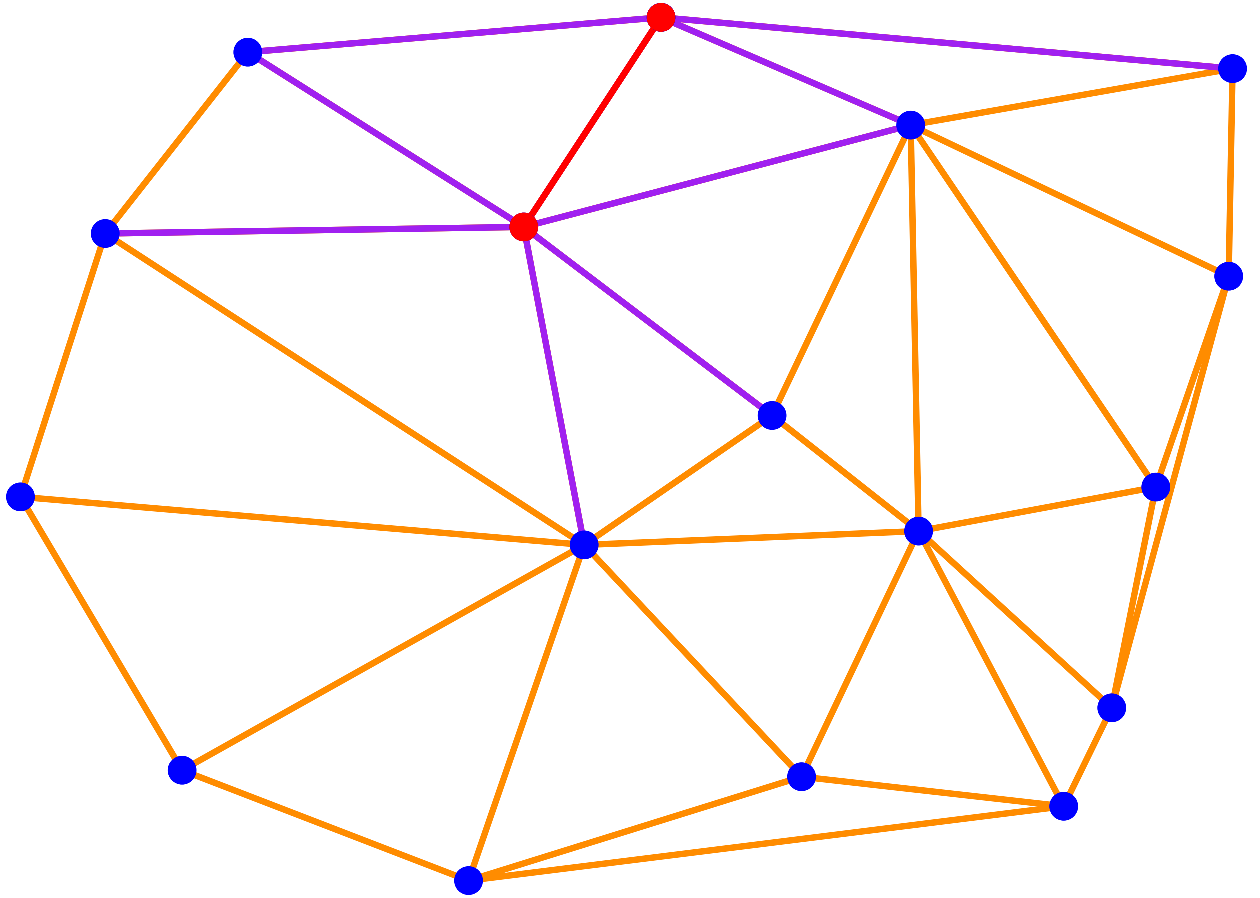
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

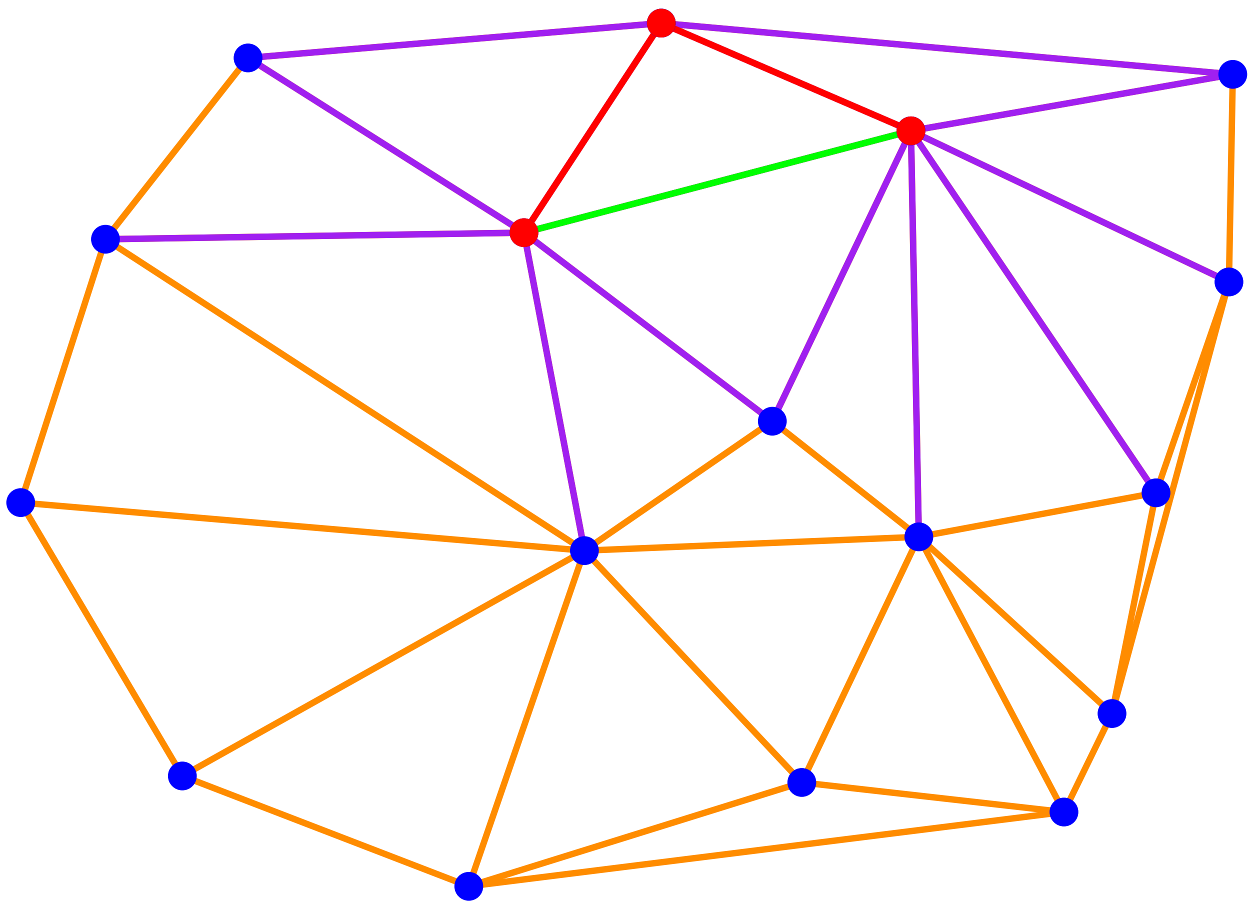
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

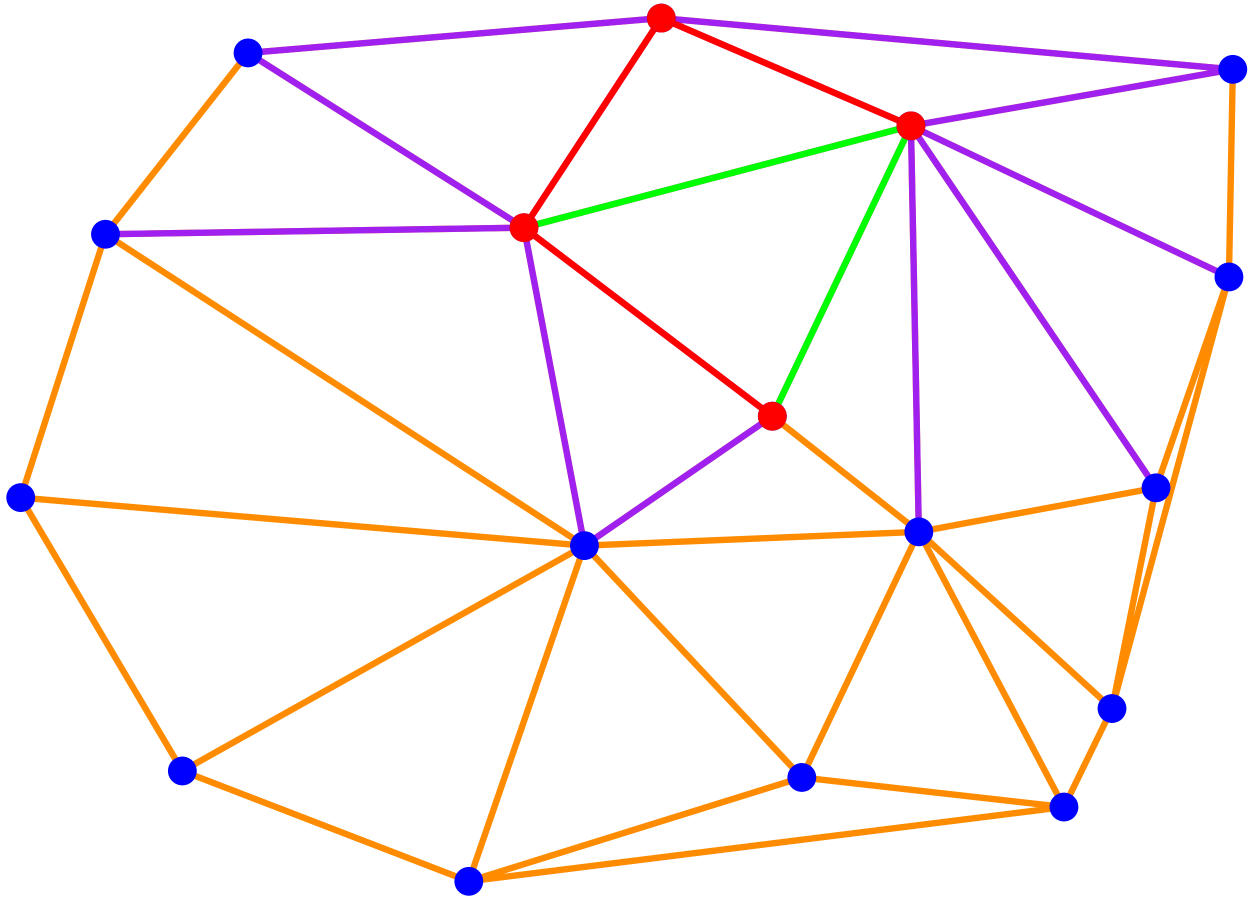
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

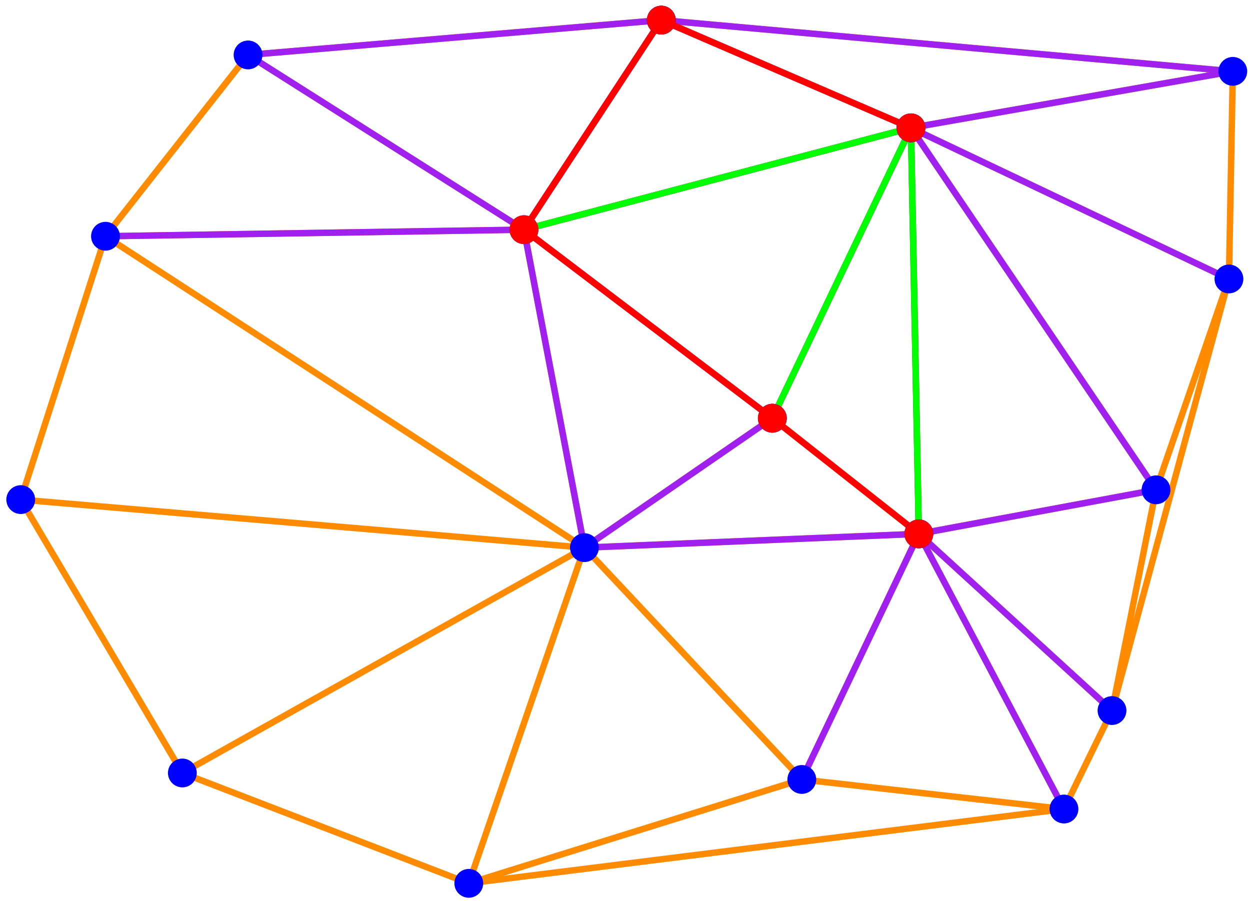
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

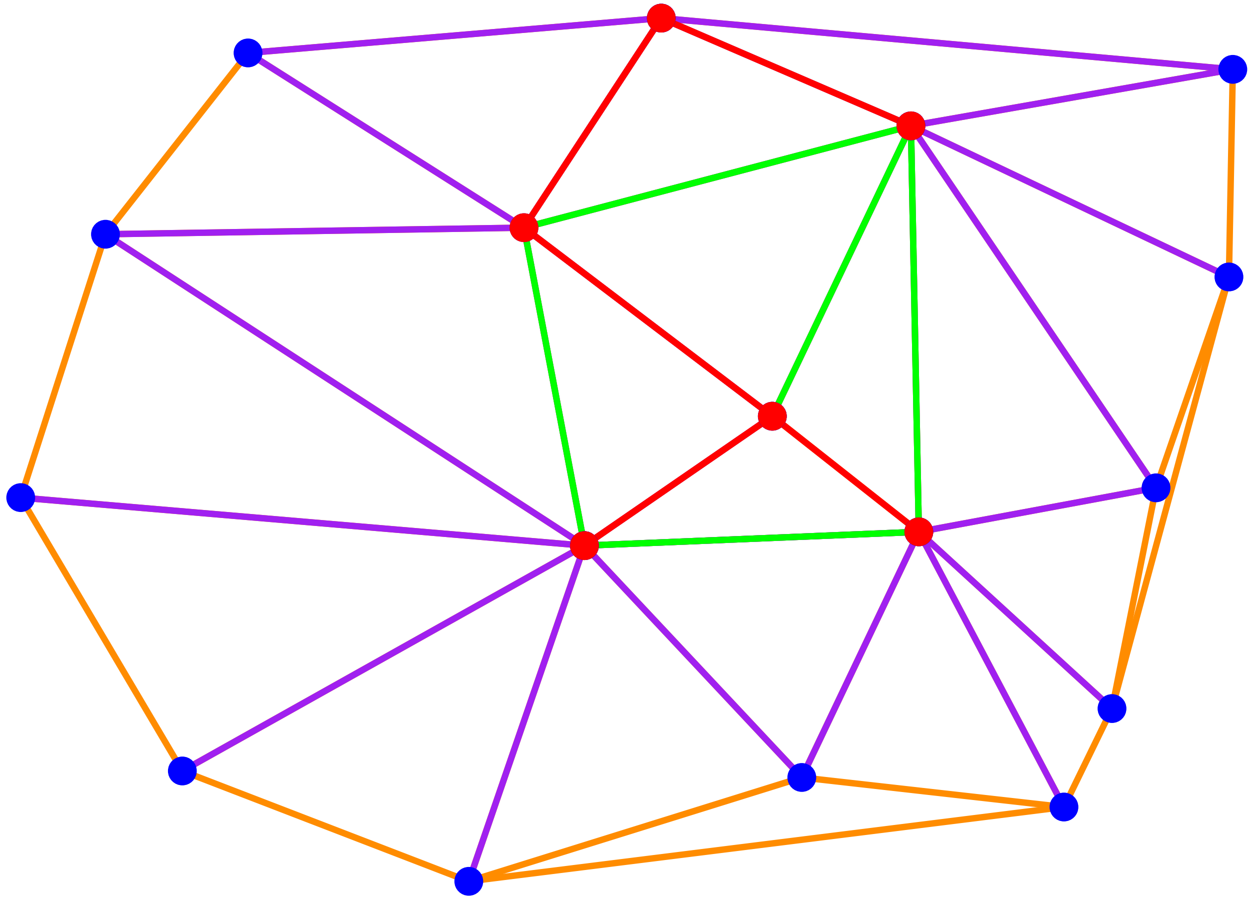
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

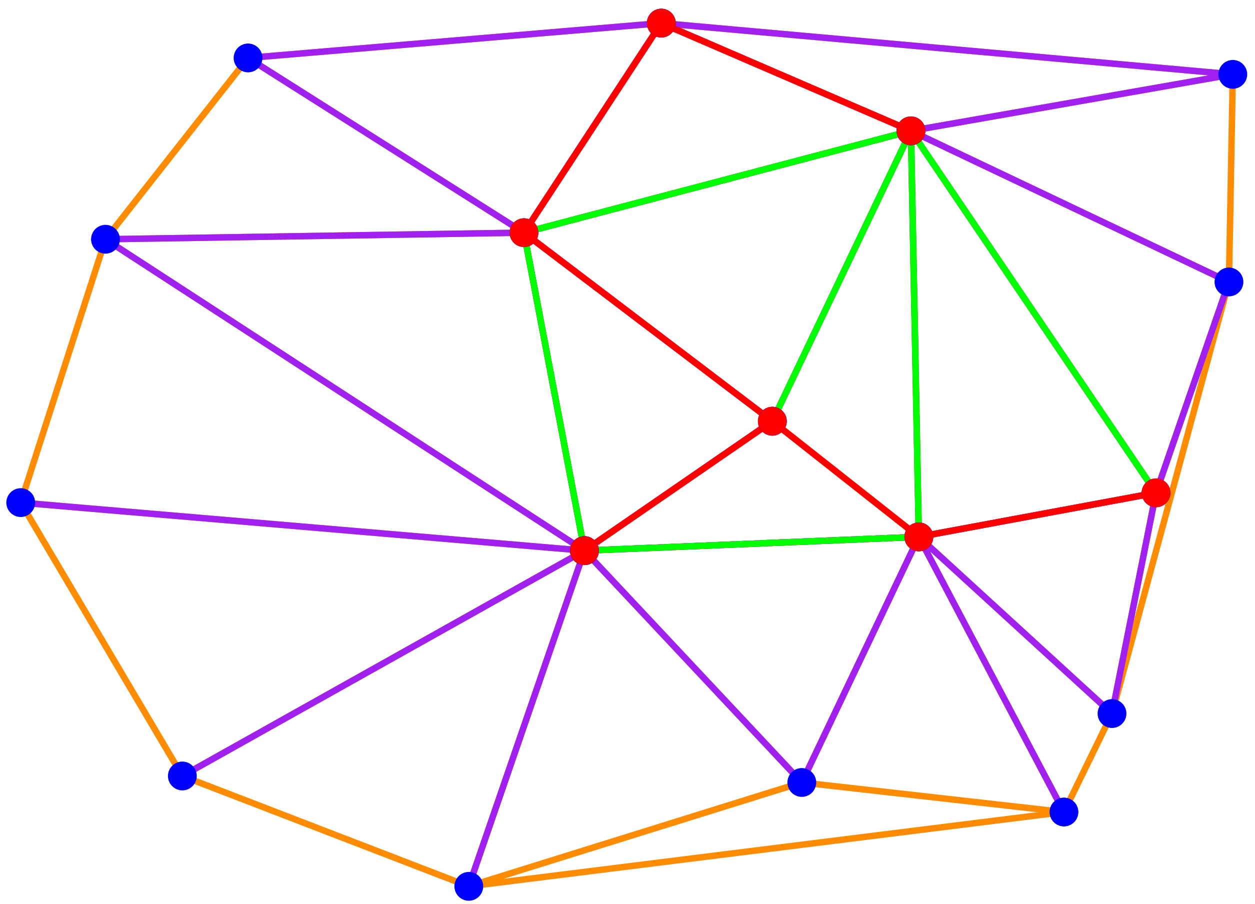
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

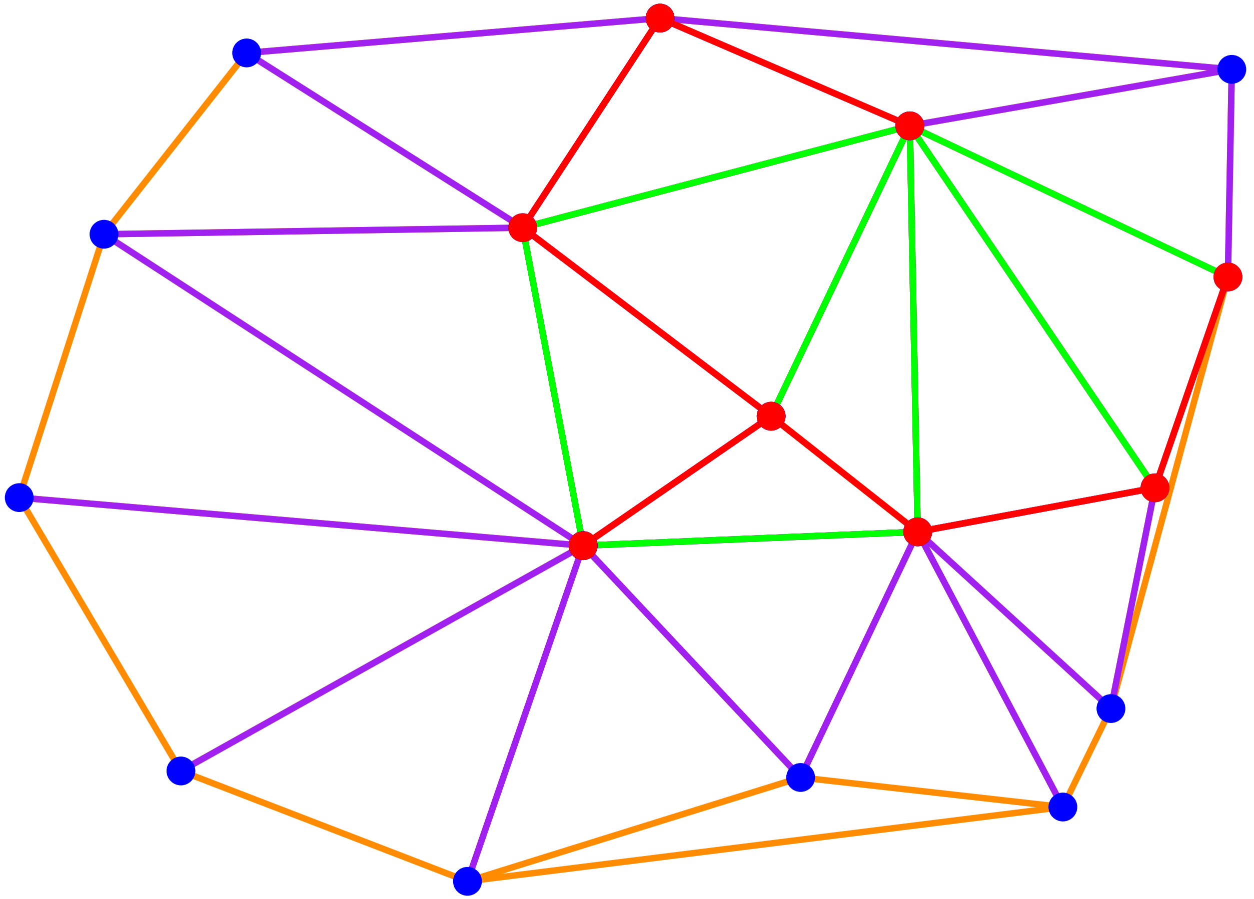
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

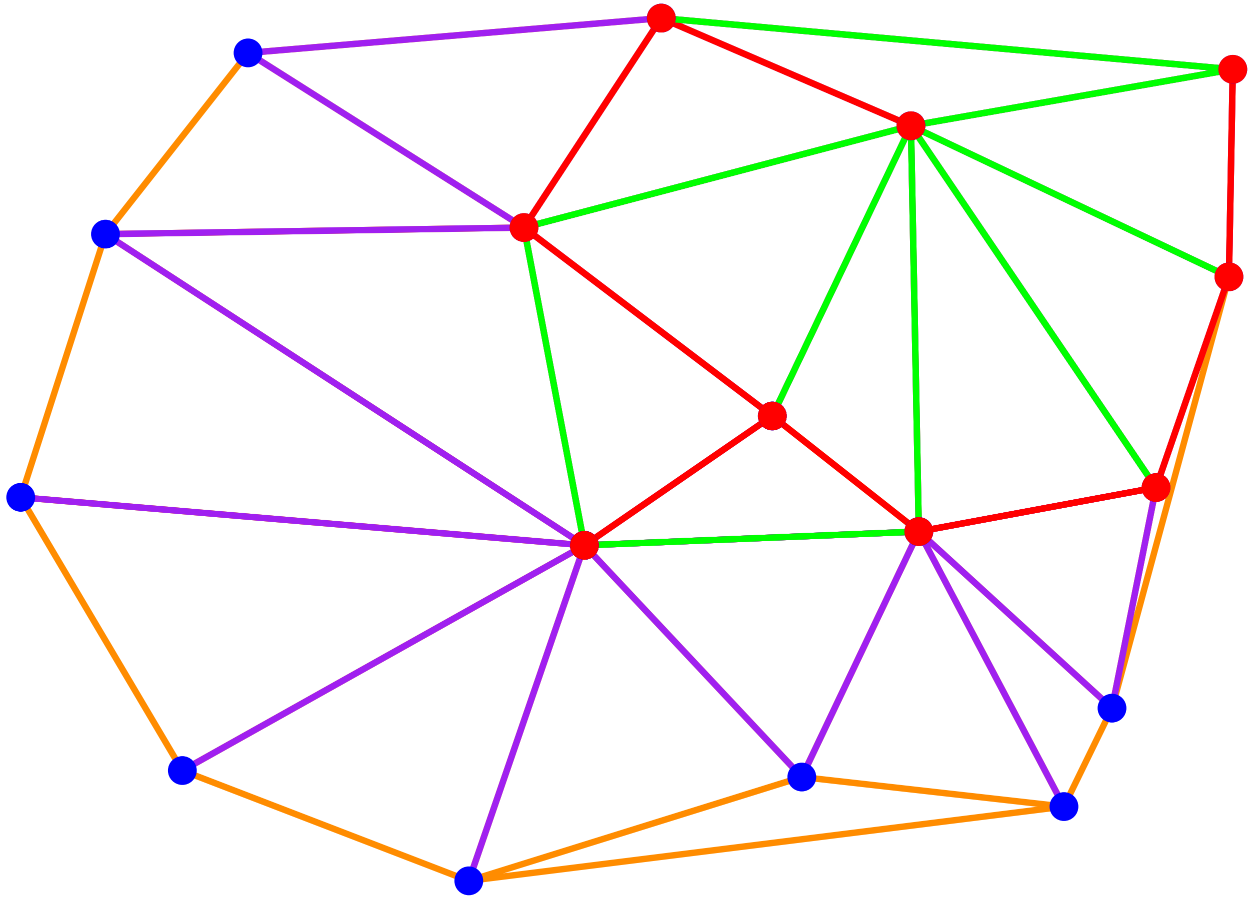
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

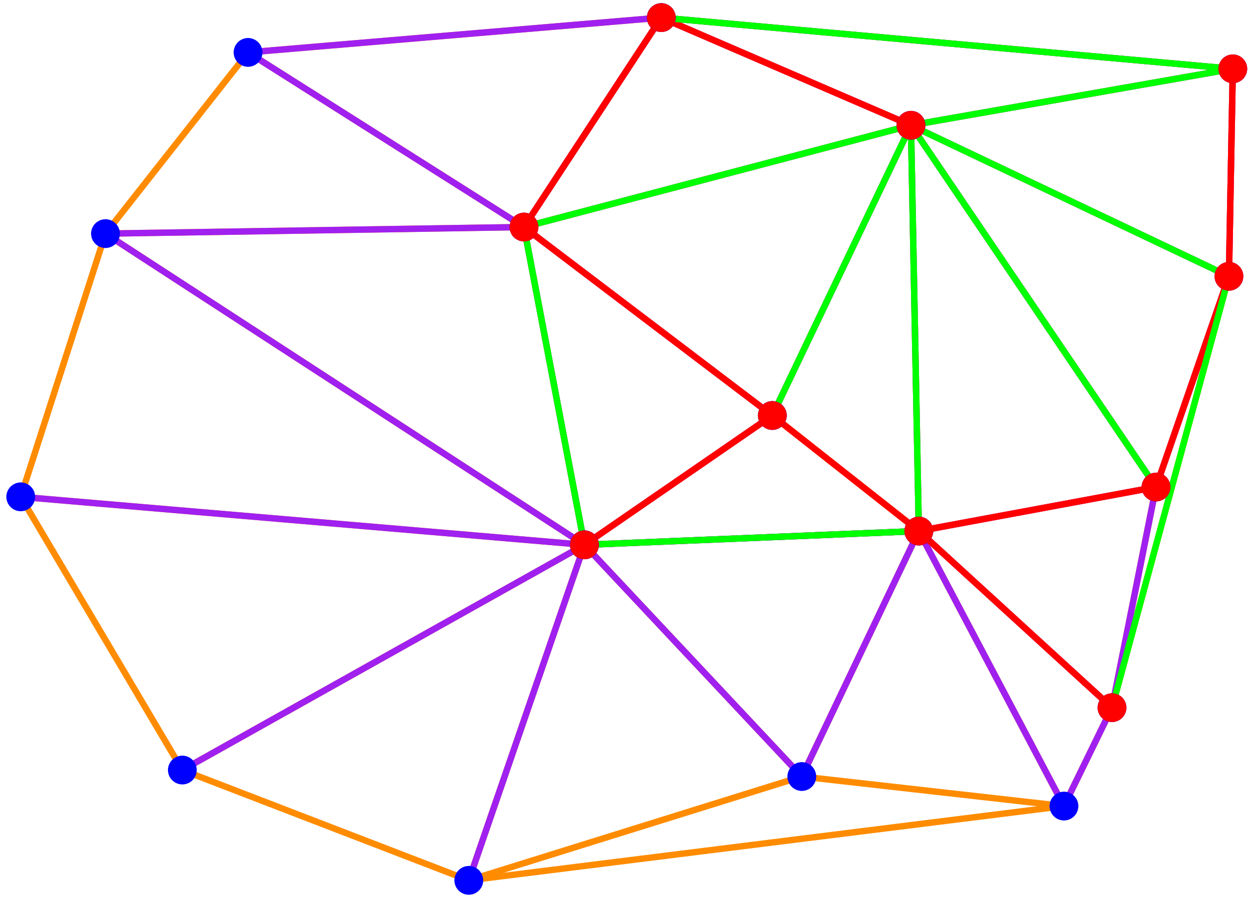
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

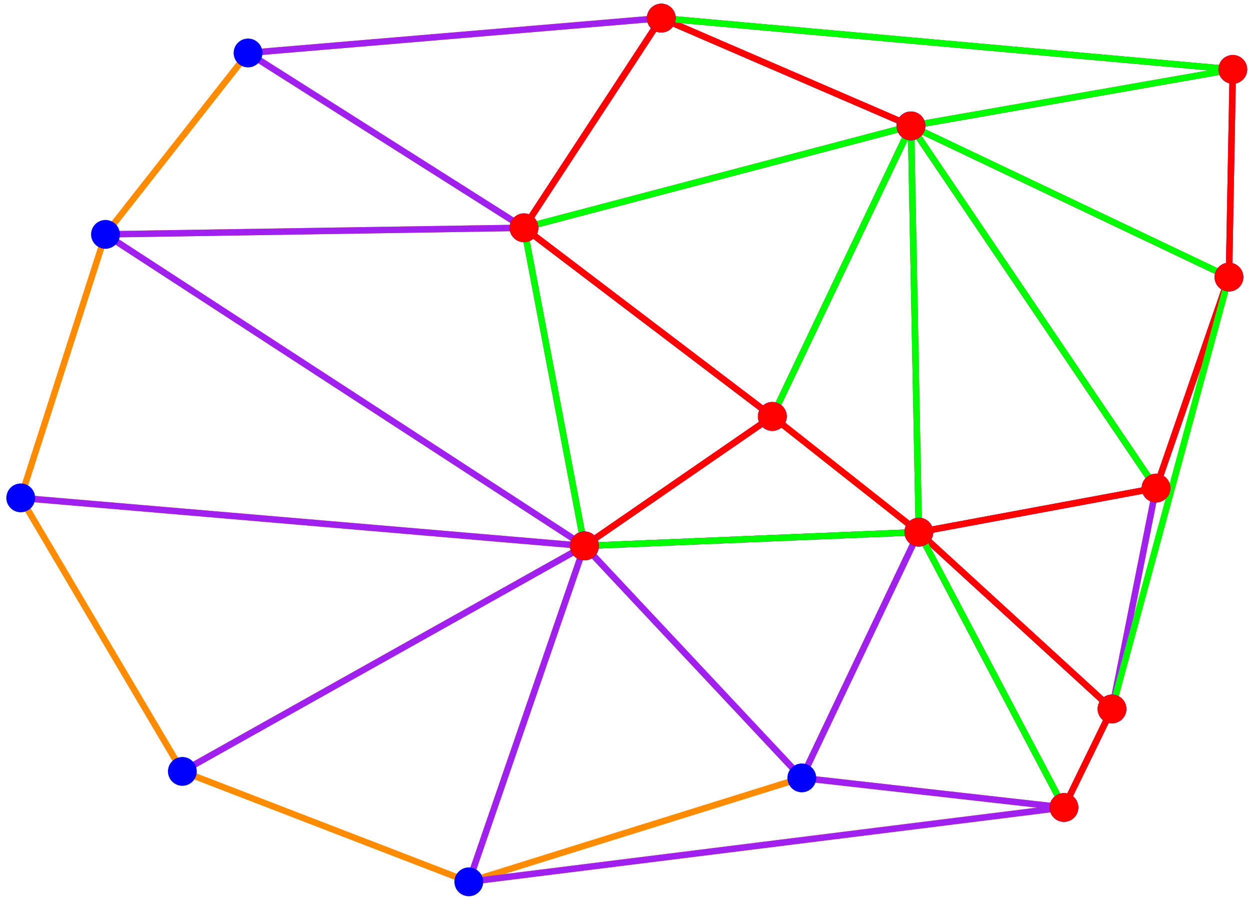
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

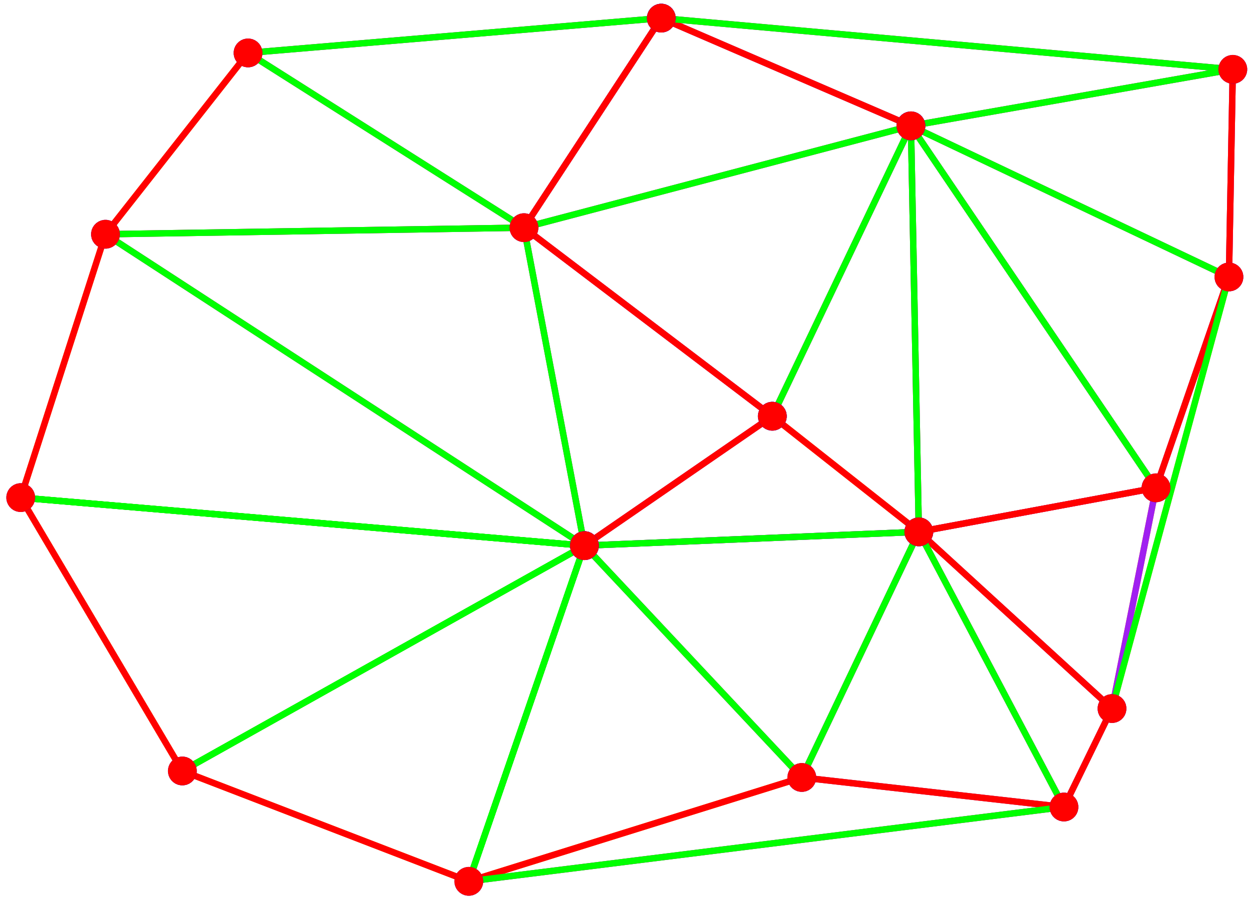
choose shorter purple edge



Delaunay Triangulation: EMST

Algorithm

choose shorter purple edge



$O(n \log n)$ after Delaunay

~~Delaunay~~ Triangulation: size

~~Delaunay~~ Triangulation: size

Convex hull

Three dimensions

Euler relation

Polytope boundary

Vertices Edges Faces

$$n \quad - \quad e \quad + \quad f \qquad = 2$$

triangular faces $\qquad 3f = 2e$

$$f = 2n - 4$$

$$e = 3n - 6$$

~~Delaunay~~ Triangulation: size

Vertices Edges Faces

$$n \quad - \quad e \quad + \quad f \quad = 2$$

~~Delaunay~~ Triangulation: size

Convex hull

k

$n - k$

Vertices Edges Faces

$$n \; - \; e \; + \; f \; = 2$$

∞

1

Triangles

t

~~Delaunay~~ Triangulation: size

Convex hull

k

$n - k$

Vertices Edges Faces

$$n \quad - \quad e \quad + \quad f \quad = 2$$

∞

Triangles

$1 \qquad t$

$$n - e + t + 1 = 2$$

~~Delaunay~~ Triangulation: size

Convex hull

k

$n - k$

Vertices Edges Faces

$$n \quad - \quad e \quad + \quad f \qquad = 2$$

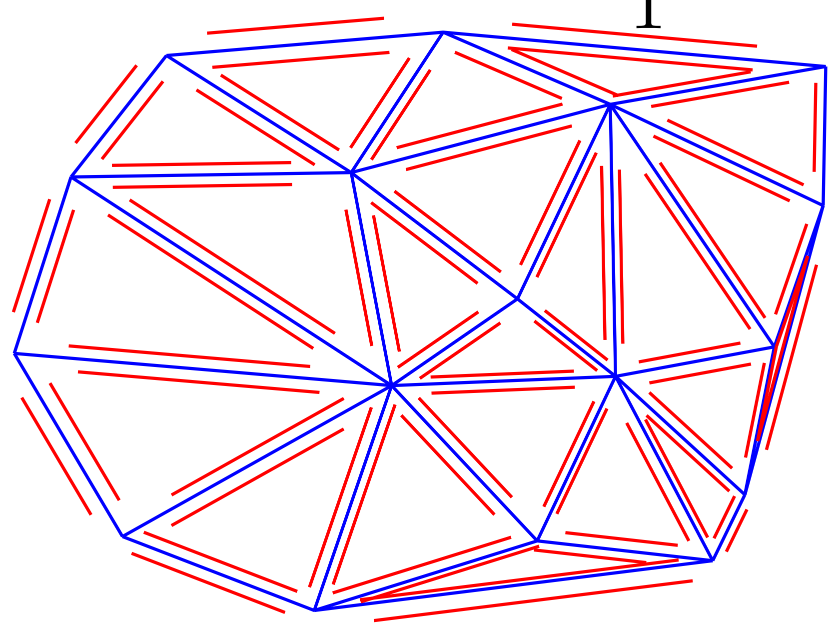
∞

Triangles

1 $\quad$ t

$$n - e + t + 1 = 2$$

$$3t + k = 2e$$



~~Delaunay~~ Triangulation: size

Convex hull

k

$n-k$

Vertices Edges Faces

$$n - e + f = 2$$

∞

1

t

Triangles

$$n - e + t + 1 = 2$$

$$3t + k = 2e \qquad\qquad 2n - 3t - k + 2t = 2$$

~~Delaunay~~ Triangulation: size

Convex hull

k

$n - k$

Vertices Edges Faces

$$n - e + f = 2$$

∞ Triangles

1 t

$$n - e + t + 1 = 2$$

$$3t + k = 2e$$

$$2n - 3t - k + 2t = 2$$

$$t = 2n - k - 2 < 2n$$

$$e = 3n - k - 3 < 3n$$

~~Delaunay~~ Triangulation: size

$$\sum_{p \in S} d^\circ(p) = 2e = 6n - 2k - 6$$

$$\mathbb{E}(d^\circ(p)) = \frac{1}{n} \sum_{p \in S} d^\circ(p) < 6$$

average on the choice of point p in set of points S

$$n - e + t + 1 = 2$$

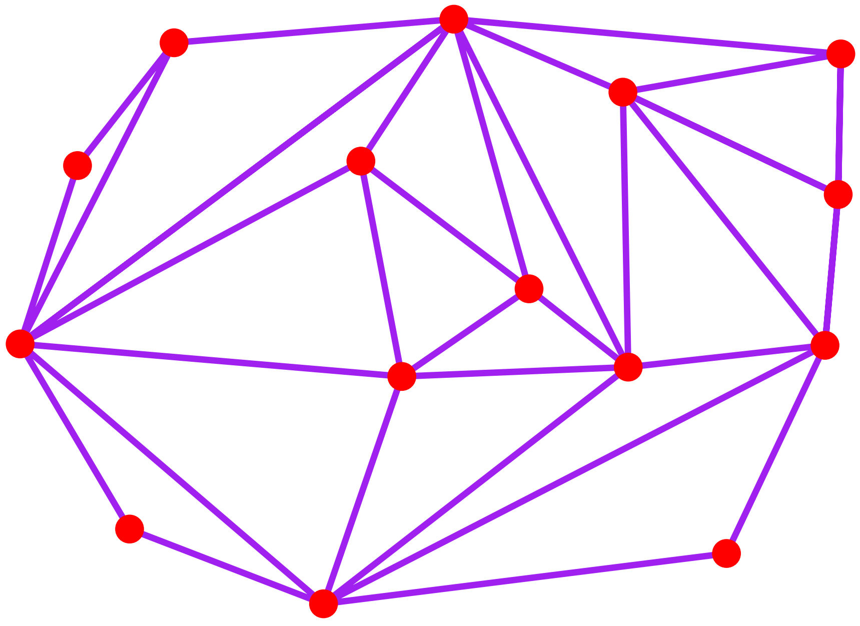
$$3t + k = 2e \qquad 2n - 3t - k + 2t = 2$$

$$t = 2n - k - 2 < 2n$$

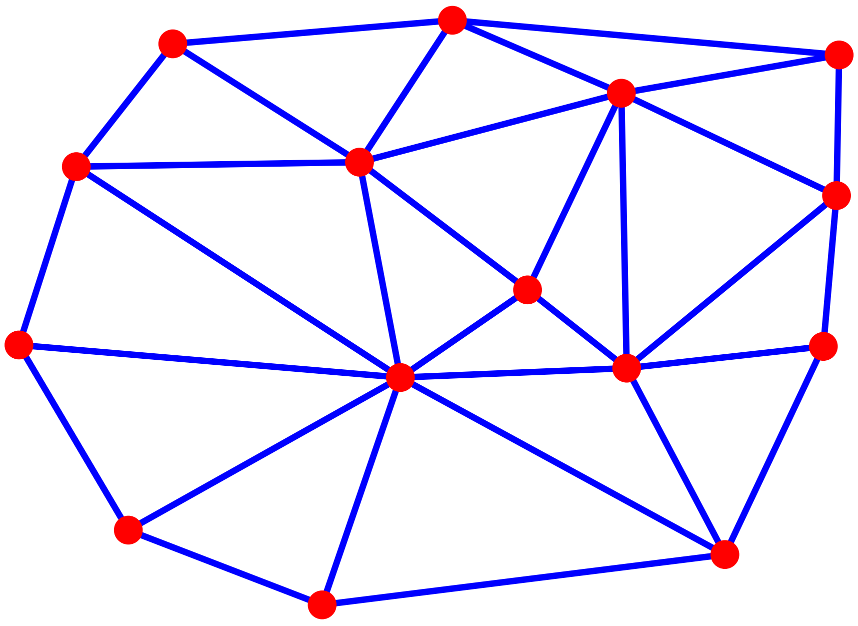
$$e = 3n - k - 3 < 3n$$

Delaunay Triangulation: max-min angle

Delaunay Triangulation: max-min angle

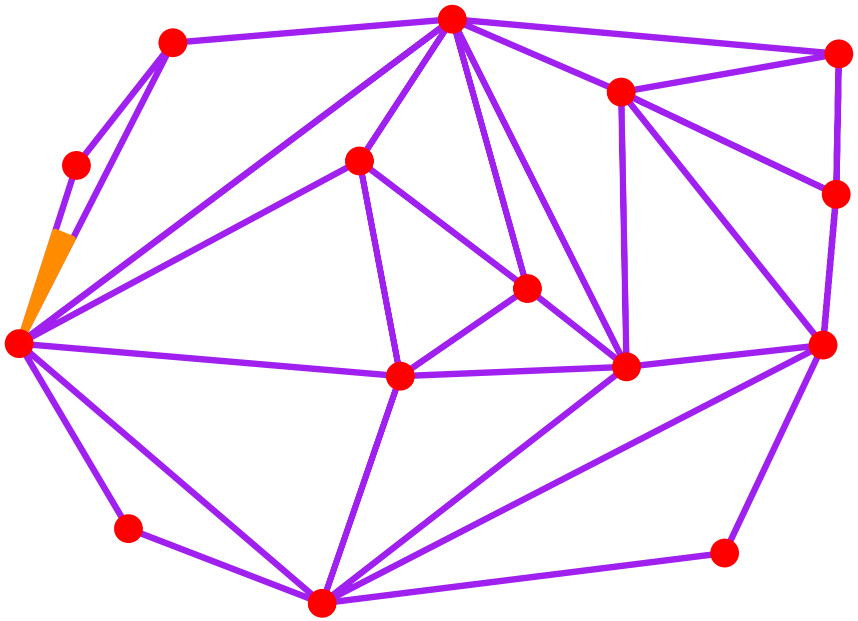


Triangulation

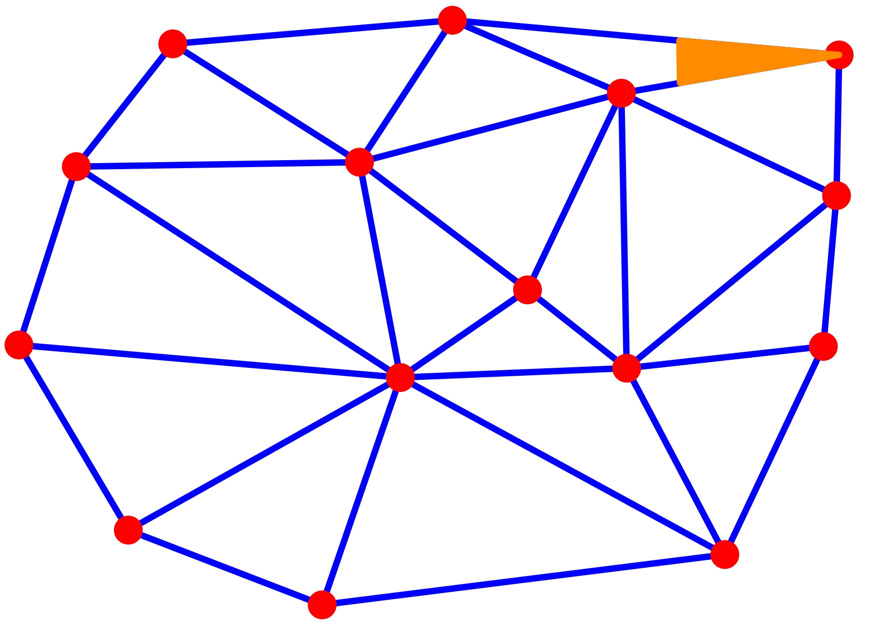


Delaunay

Delaunay Triangulation: max-min angle



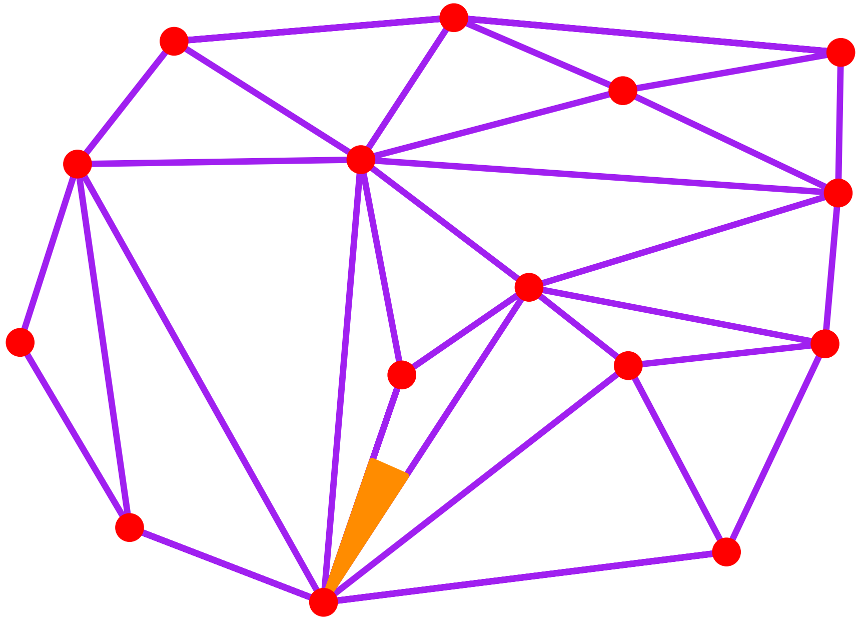
Triangulation



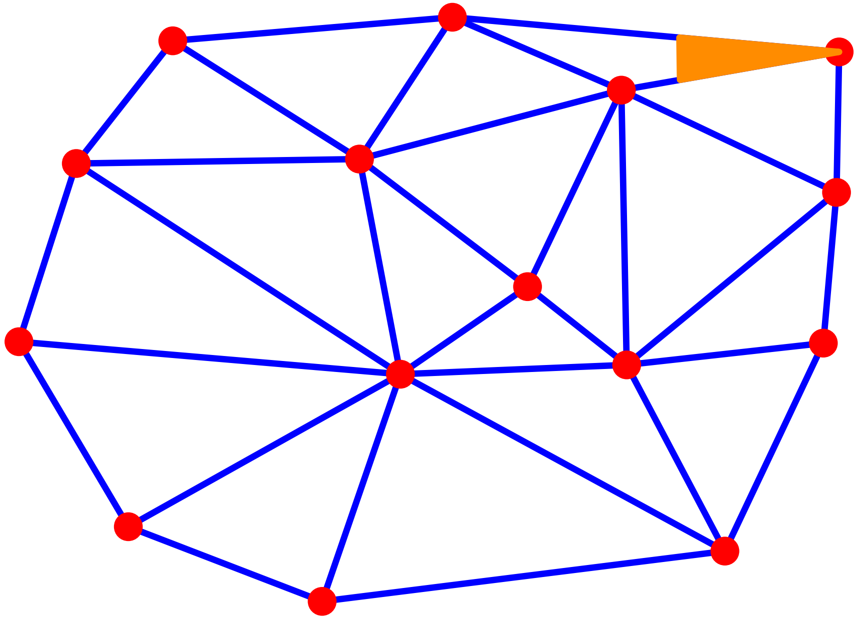
Delaunay

smallest angle

Delaunay Triangulation: max-min angle



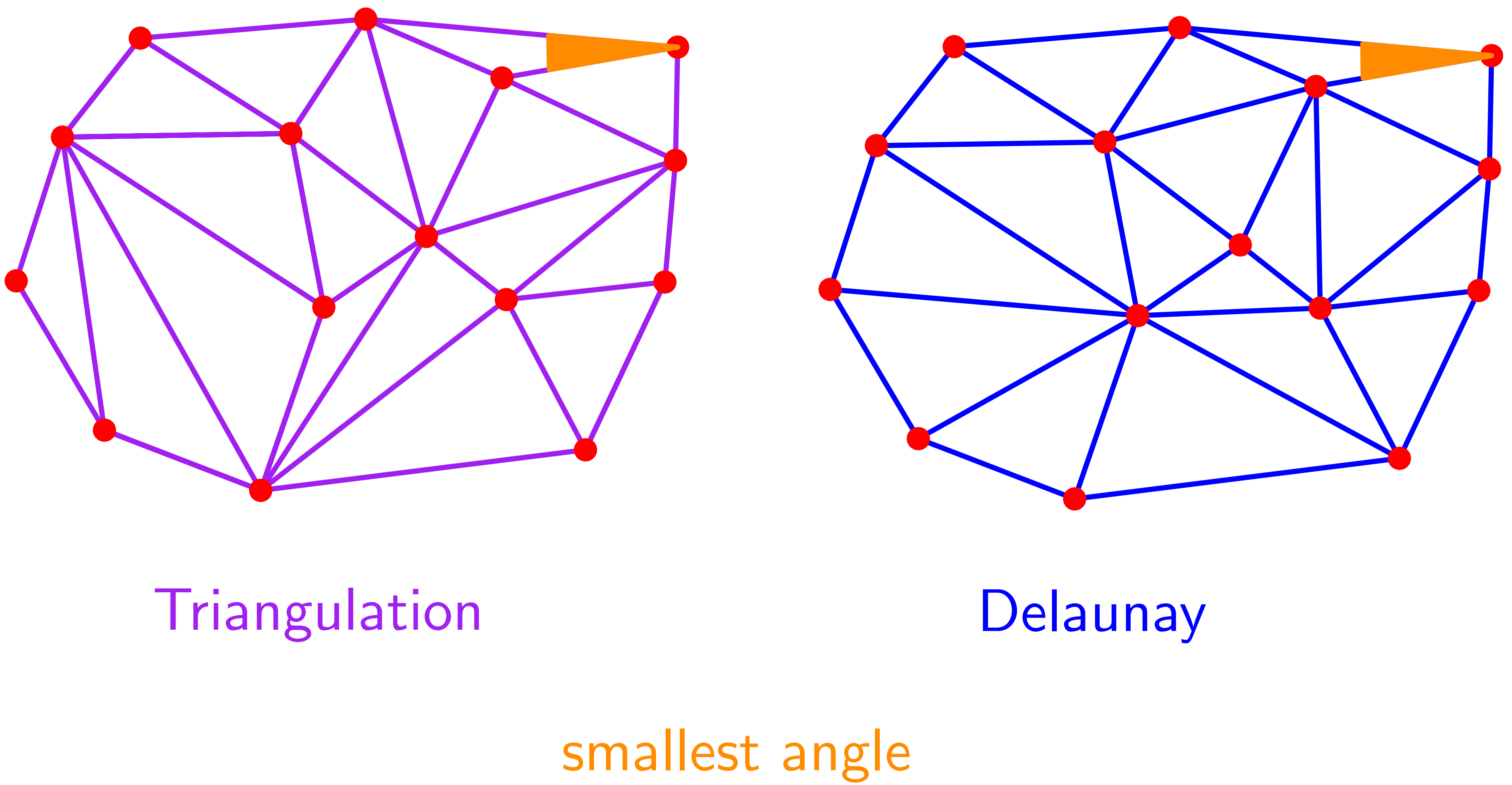
Triangulation



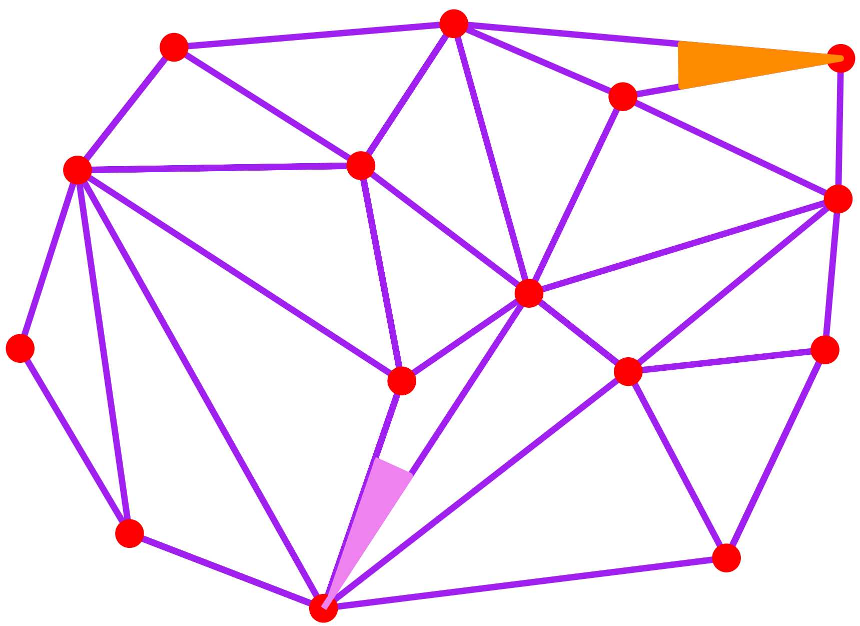
Delaunay

smallest angle

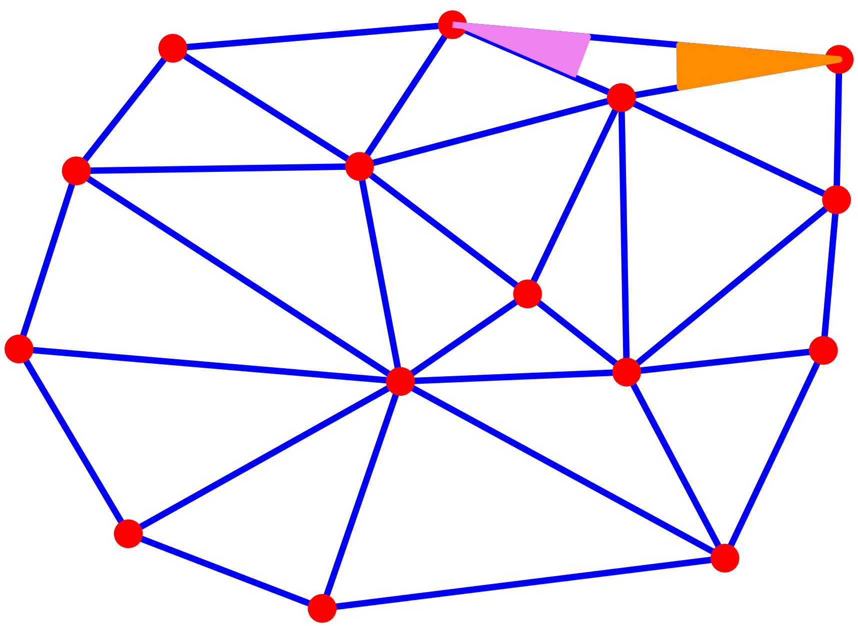
Delaunay Triangulation: max-min angle



Delaunay Triangulation: max-min angle



Triangulation



Delaunay

smallest angle

second smallest angle

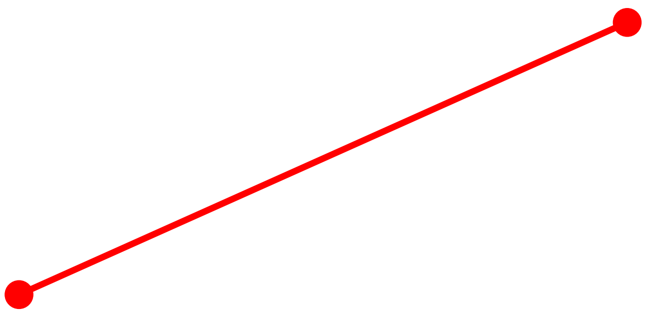
Delaunay Triangulation: max-min angle

Proof

Delaunay Triangulation: max-min angle

Definition

Delaunay edge

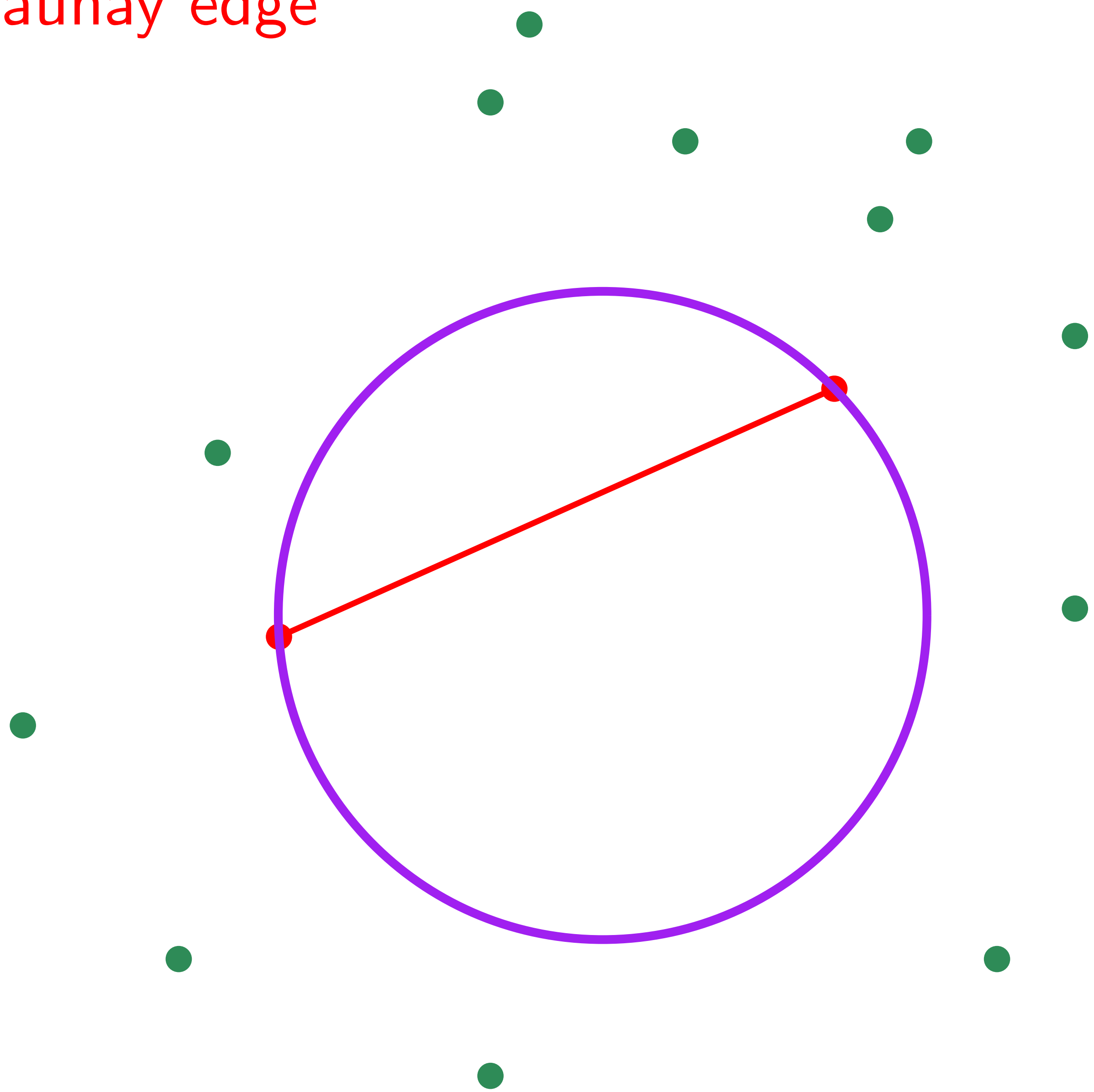


Delaunay Triangulation: max-min angle

Definition

Delaunay edge

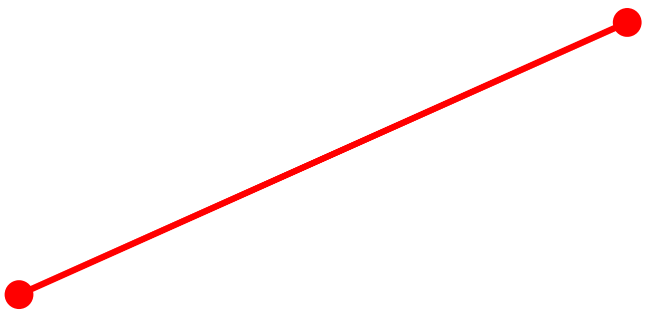
$\exists$ empty circle



Delaunay Triangulation: max-min angle

Definition

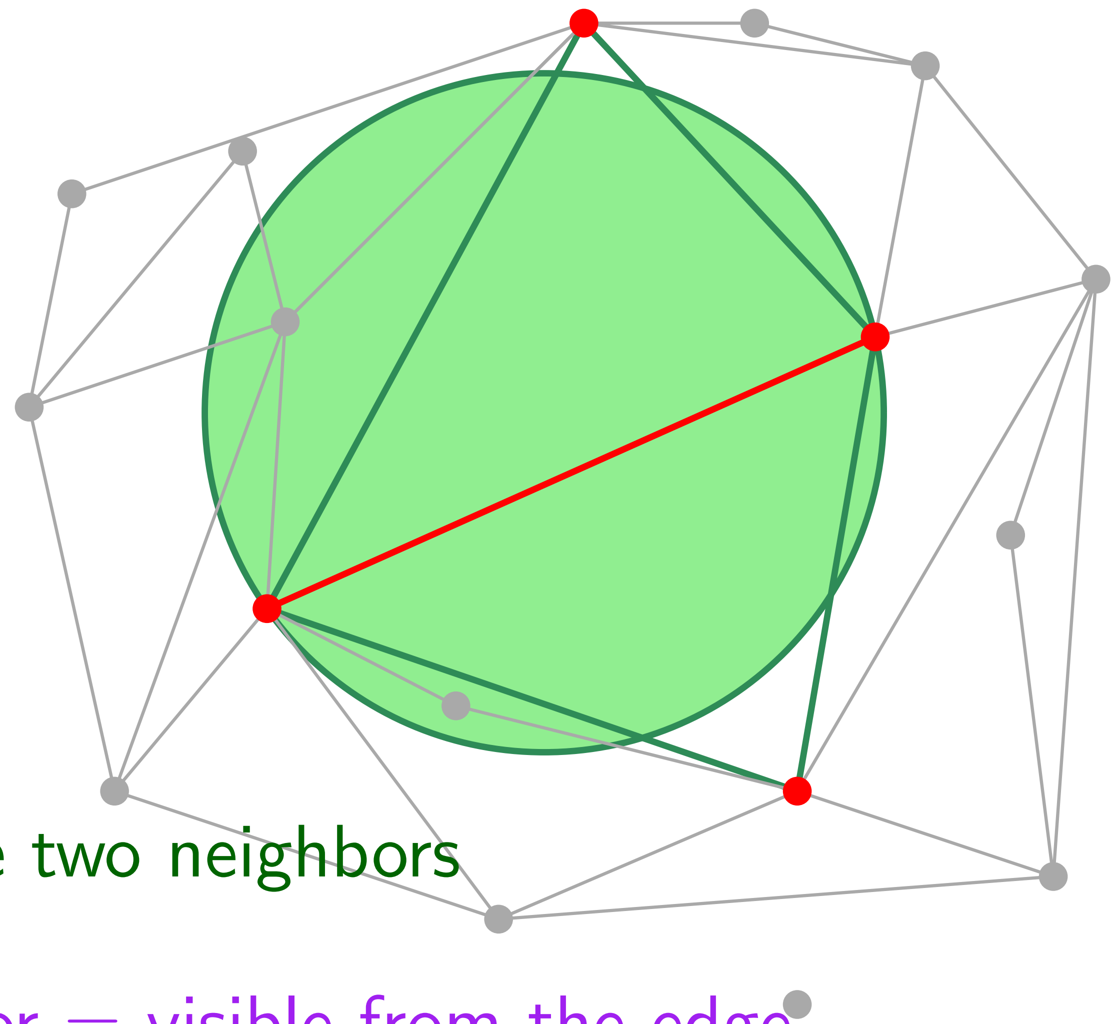
locally Delaunay edge w.r.t. a triangulation



Delaunay Triangulation: max-min angle

Definition

locally Delaunay edge w.r.t. a triangulation



$\exists$ circle

not enclosing the two neighbors

neighbor = visible from the edge

Delaunay Triangulation: max-min angle

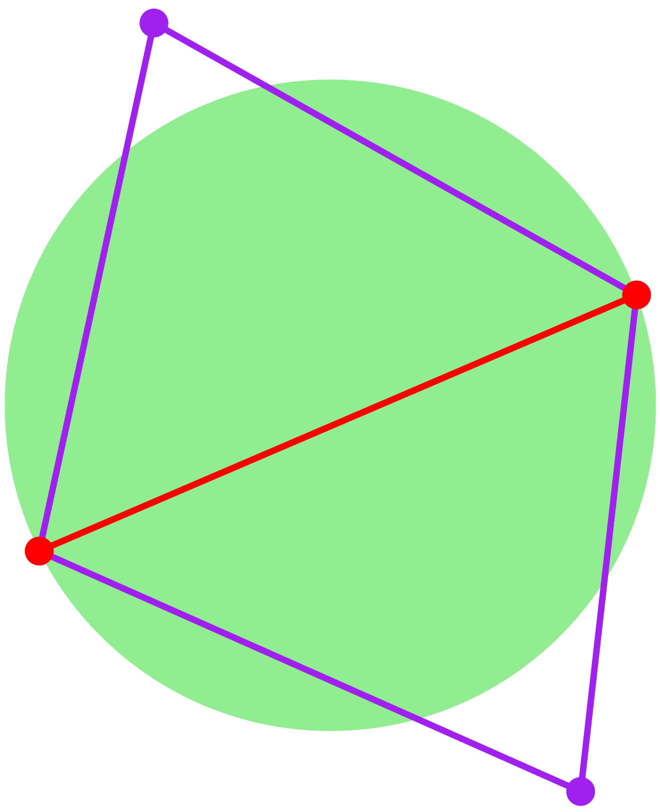
Lemma $\quad (\forall$ edge: locally Delaunay$) \iff$ Delaunay

Delaunay Triangulation: max-min angle

Lemma (∀ edge: locally Delaunay) $\iff$ Delaunay

Proof:

choose an edge

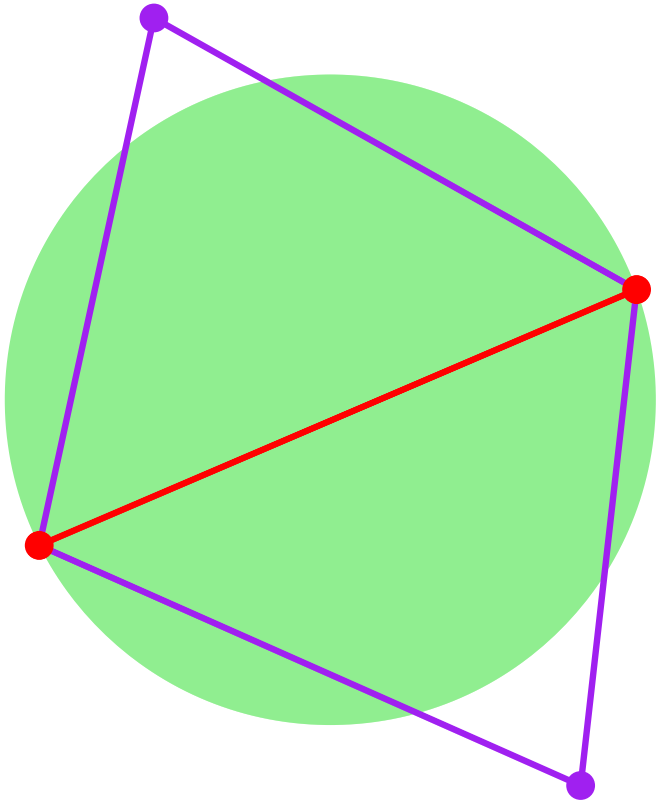


Delaunay Triangulation: max-min angle

Lemma (∀ edge: locally Delaunay) $\iff$ Delaunay

Proof:

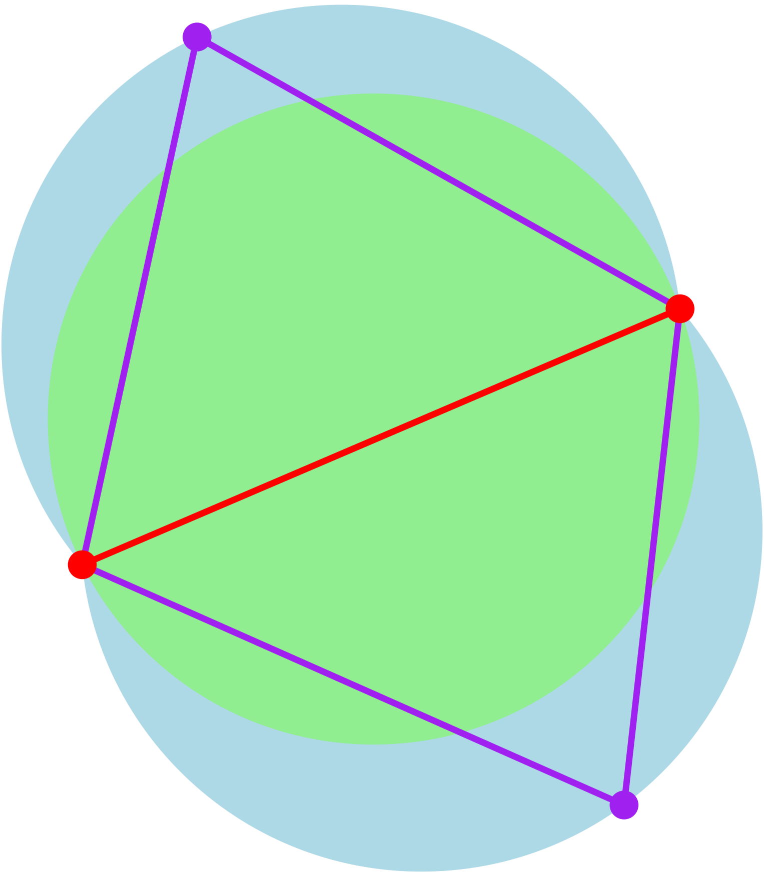
- choose an edge
- edges of the quadrilateral are locally Delaunay



Delaunay Triangulation: max-min angle

Lemma (∀ edge: locally Delaunay) $\Longleftrightarrow$ Delaunay

Proof:

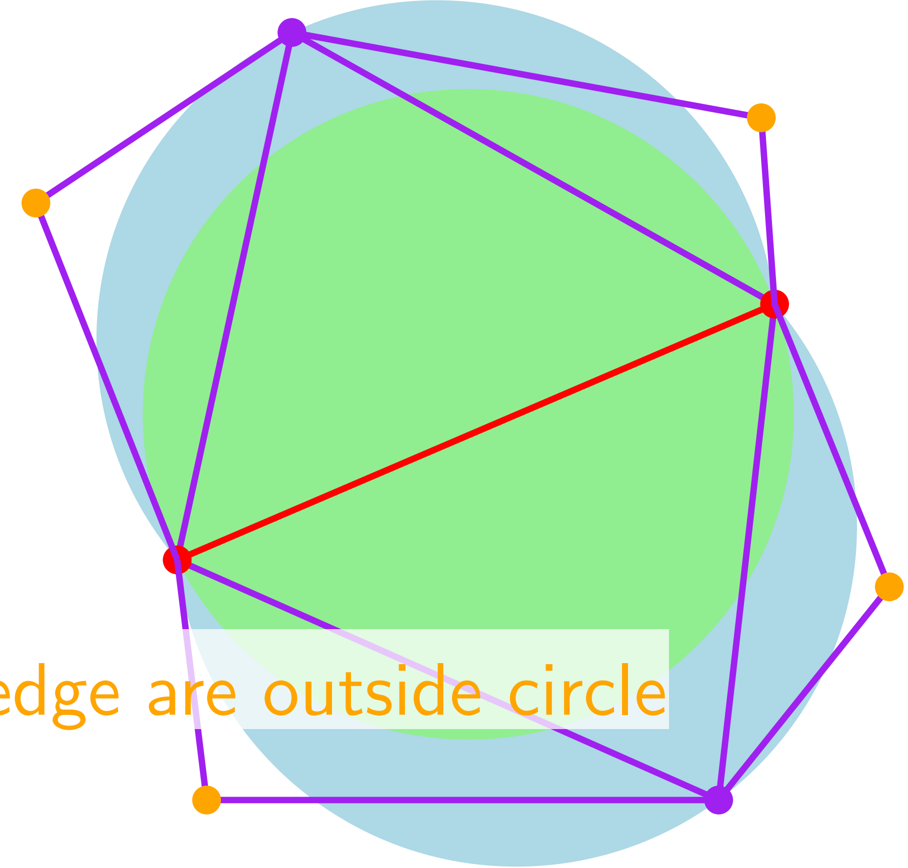


- choose an edge
- edges of the quadrilateral are locally Delaunay

Delaunay Triangulation: max-min angle

Lemma ($\forall$ edge: locally Delaunay) $\Longleftrightarrow$ Delaunay

Proof:

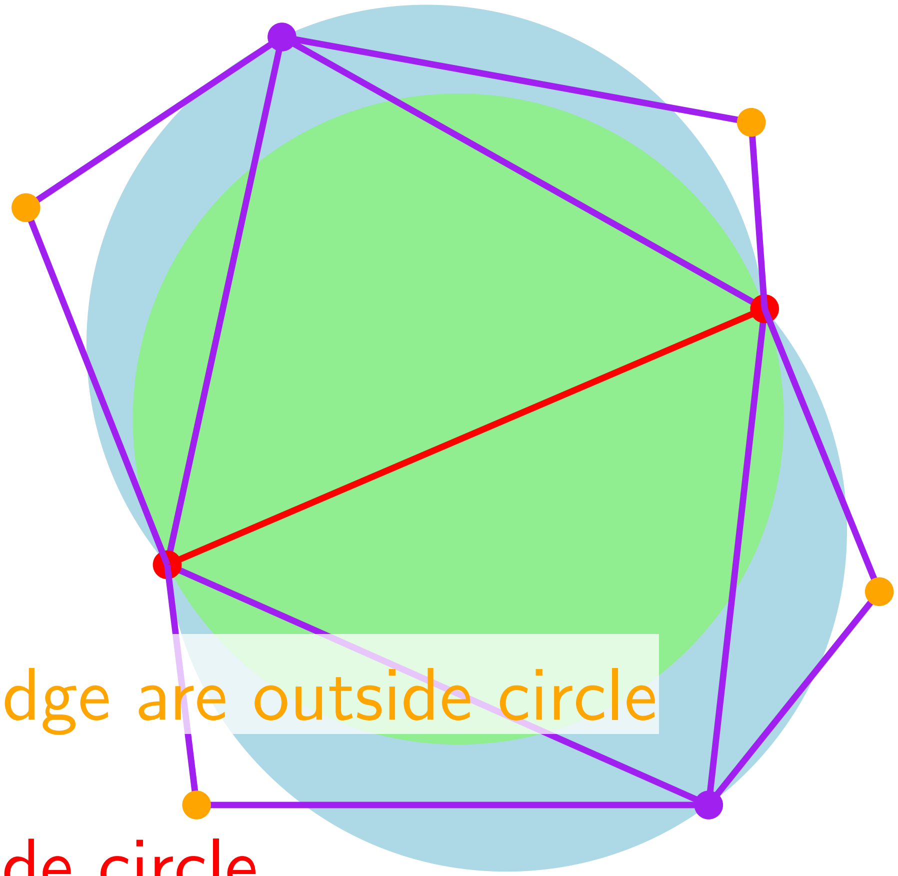


- choose an edge
- edges of the quadrilateral are locally Delaunay
- Vertices visible through one edge are outside circle

Delaunay Triangulation: max-min angle

Lemma (∀ edge: locally Delaunay) $\iff$ Delaunay

Proof:



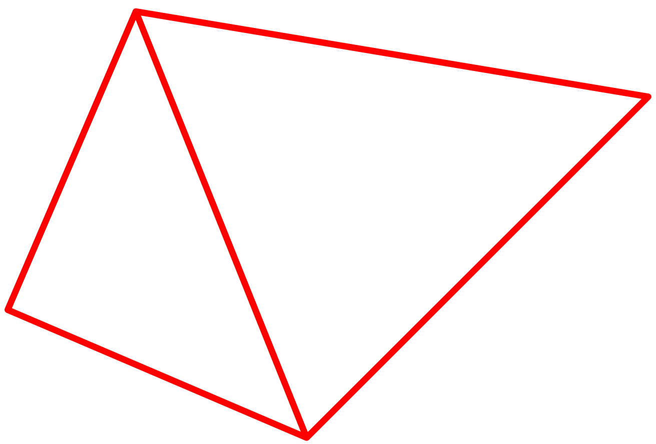
- choose an edge
- edges of the quadrilateral
  are locally Delaunay
- Vertices visible through one edge are outside circle
- Induction → all vertices outside circle

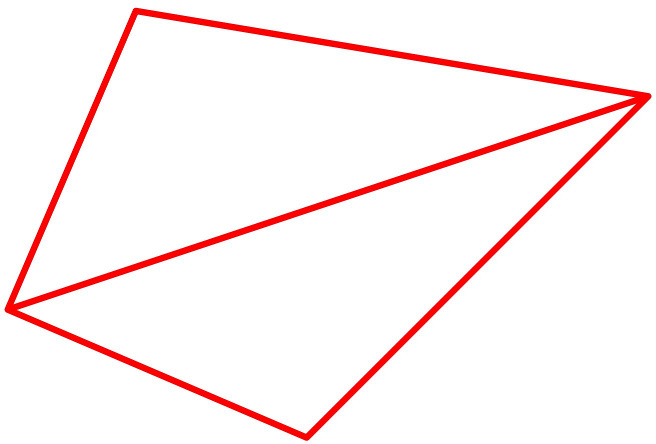
Delaunay Triangulation: max-min angle

Lemma For four points in convex position

Delaunay $\Longleftrightarrow$ maximize the smallest angle

Two possible triangulation



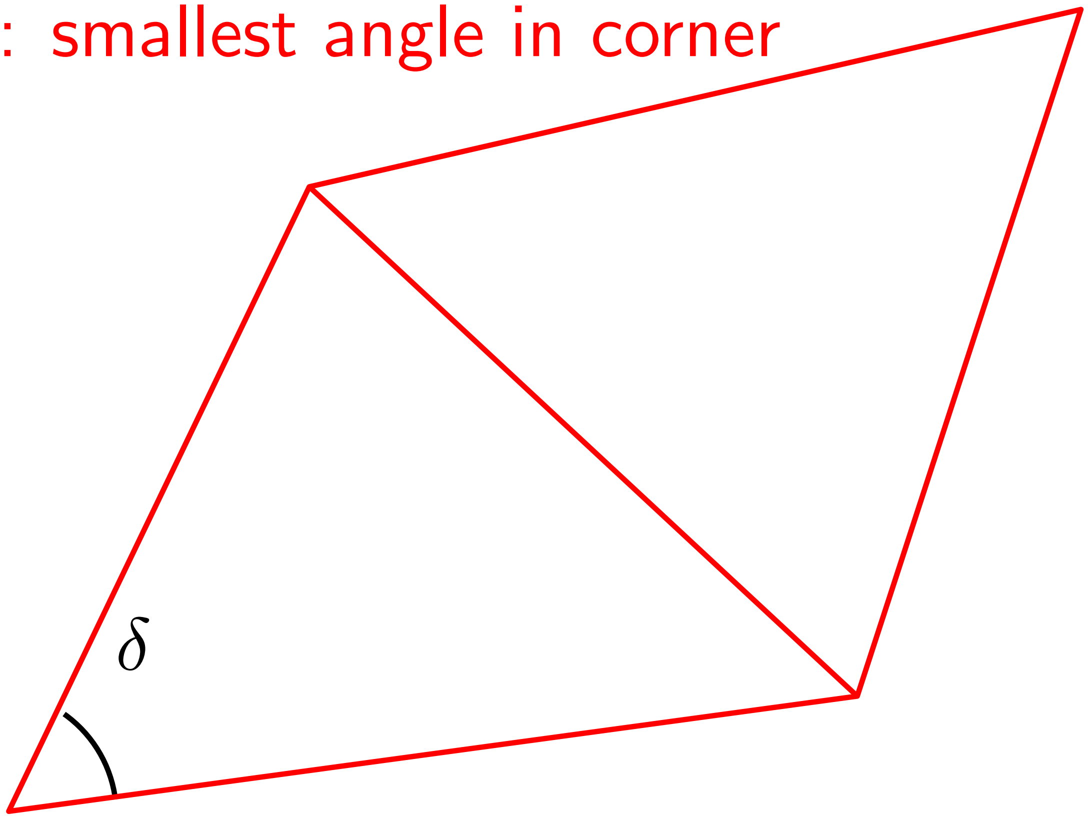


Delaunay Triangulation: max-min angle

Lemma For four points in convex position

Delaunay $\Longleftrightarrow$ maximize the smallest angle

Case 1: smallest angle in corner

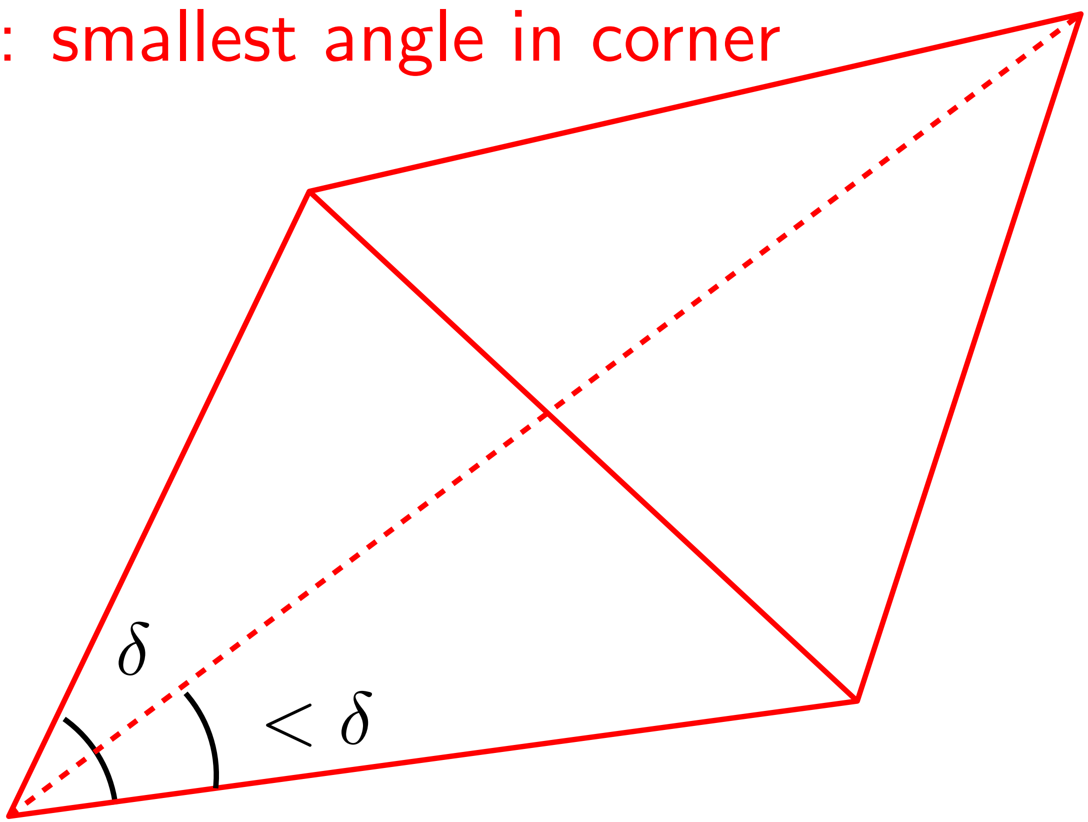


Delaunay Triangulation: max-min angle

Lemma For four points in convex position

Delaunay $\Longleftrightarrow$ maximize the smallest angle

Case 1: smallest angle in corner



$\exists$ a smaller angle $\in$ other triangulation

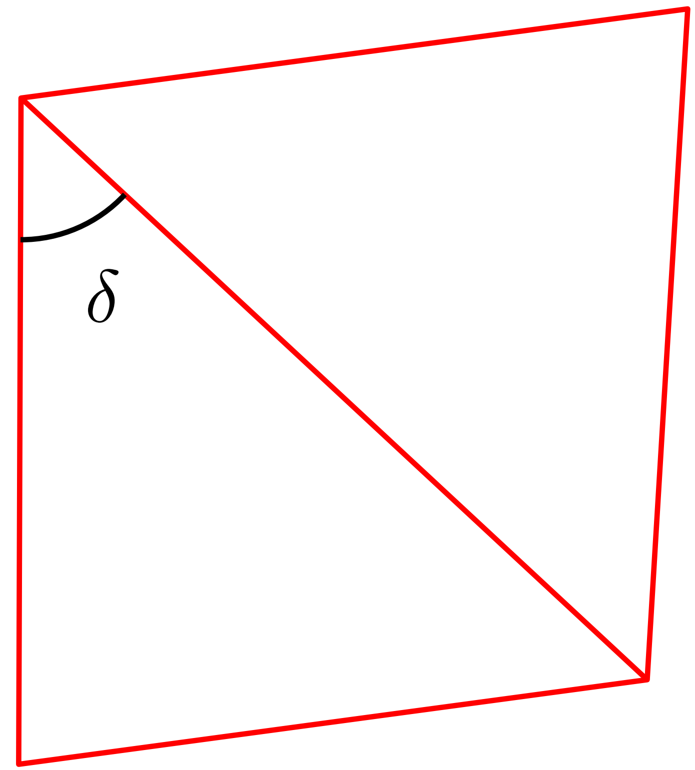
Delaunay Triangulation: max-min angle

Lemma For four points in convex position

Delaunay $\iff$ maximize the smallest angle

Case 2: smallest angle

along diagonal



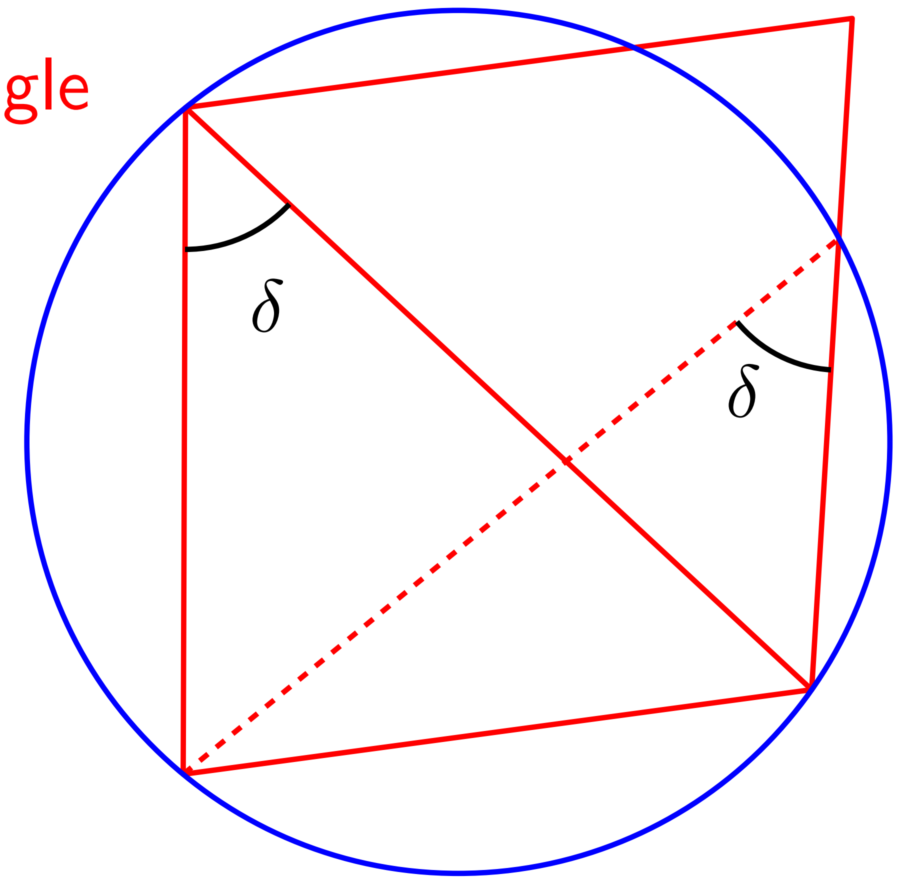
Delaunay Triangulation: max-min angle

Lemma For four points in convex position

Delaunay $\Longleftrightarrow$ maximize the smallest angle

Case 2: smallest angle

along diagonal

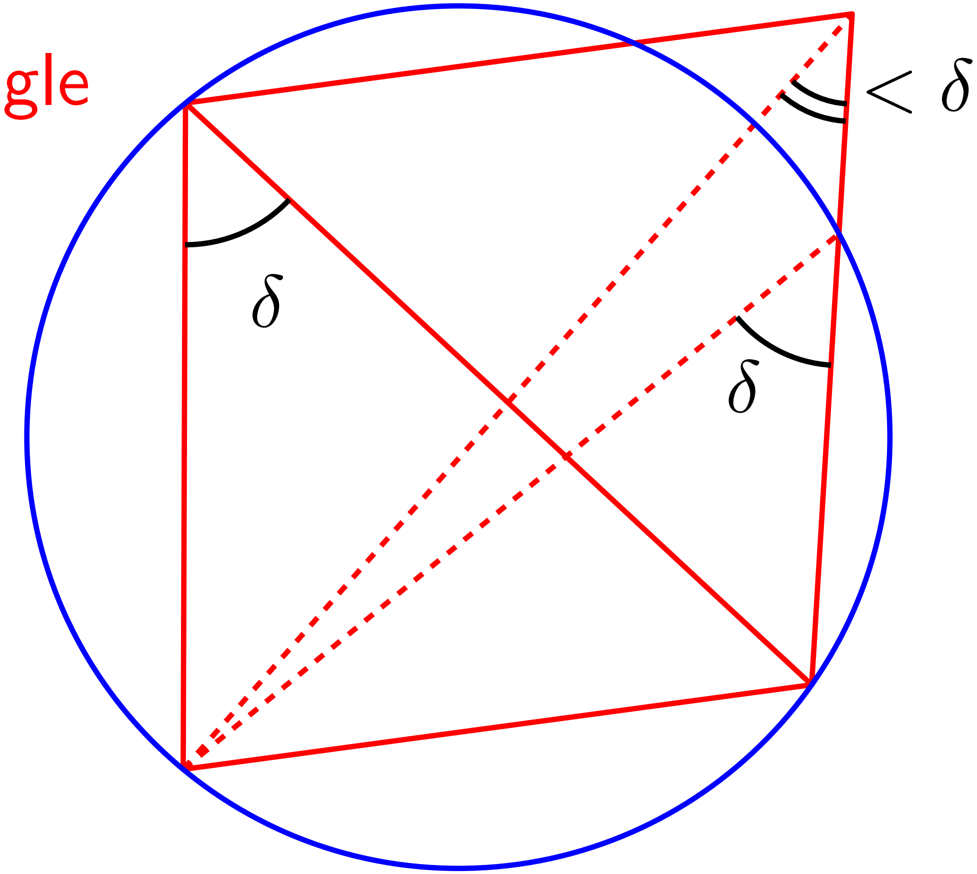


Delaunay Triangulation: max-min angle

Lemma For four points in convex position

Delaunay $\Longleftrightarrow$ maximize the smallest angle

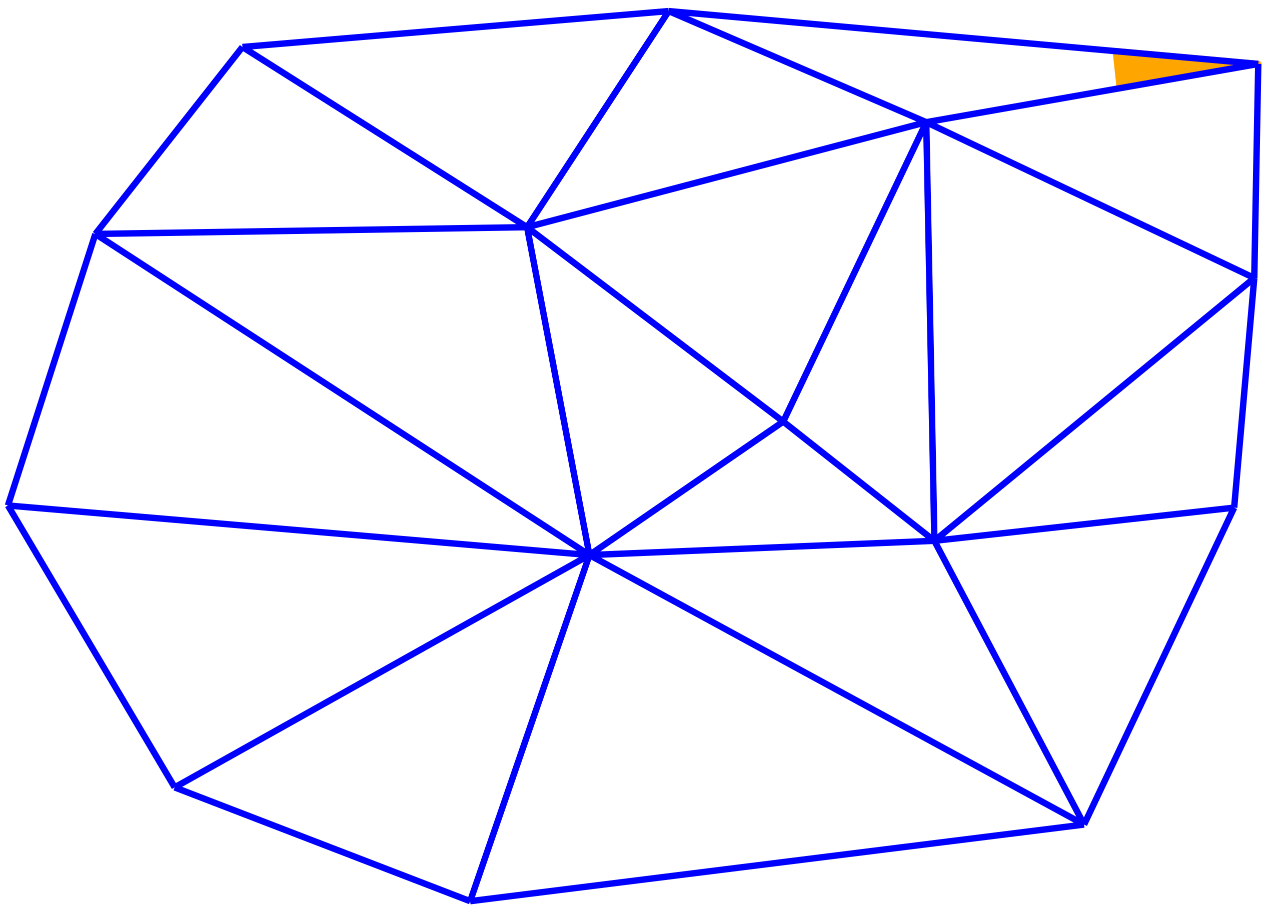
Case 2: smallest angle along diagonal



$\exists$ a smaller angle $\in$ other triangulation

Delaunay Triangulation: max-min angle

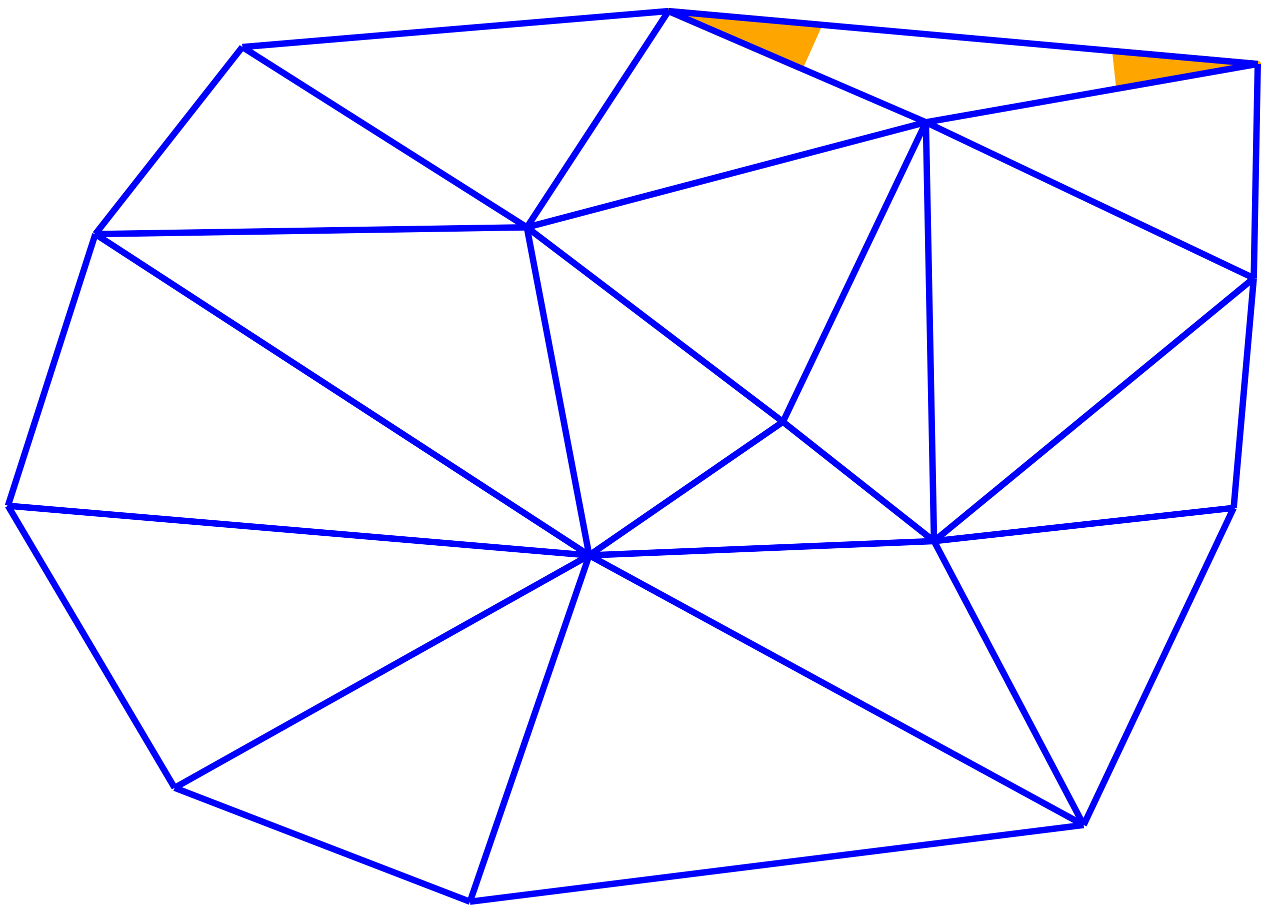
Map: Triangulations $\longrightarrow \mathbb{R}^{6n-3k-4}$ smallest angle α_1



Delaunay Triangulation: max-min angle

Map: Triangulations $\longrightarrow \mathbb{R}^{6n-3k-4}$

smallest angle α_1
second smallest angle α_2



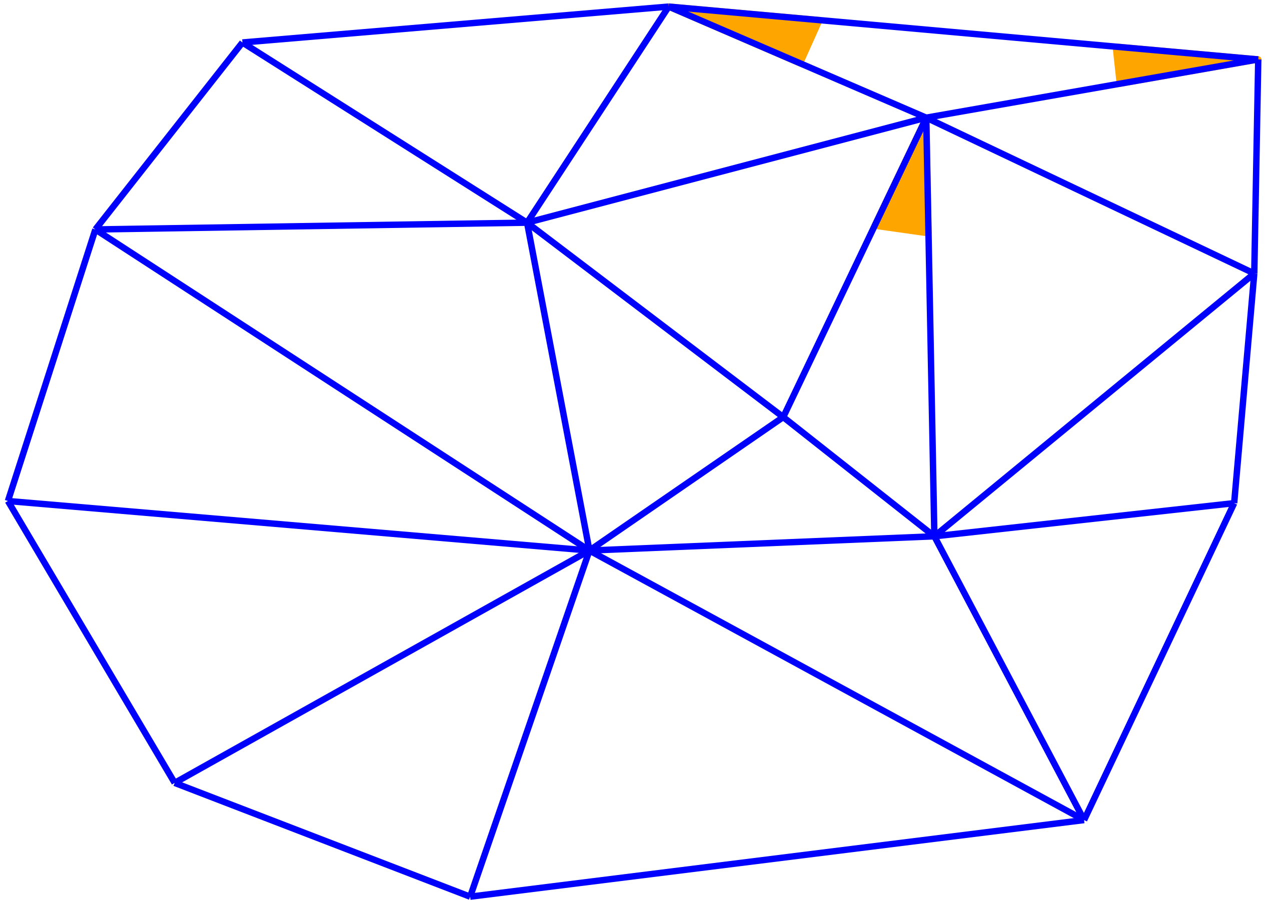
Delaunay Triangulation: max-min angle

Map: Triangulations $\longrightarrow \mathbb{R}^{6n-3k-4}$

smallest angle α_1

second smallest angle α_2

third smallest angle α_3

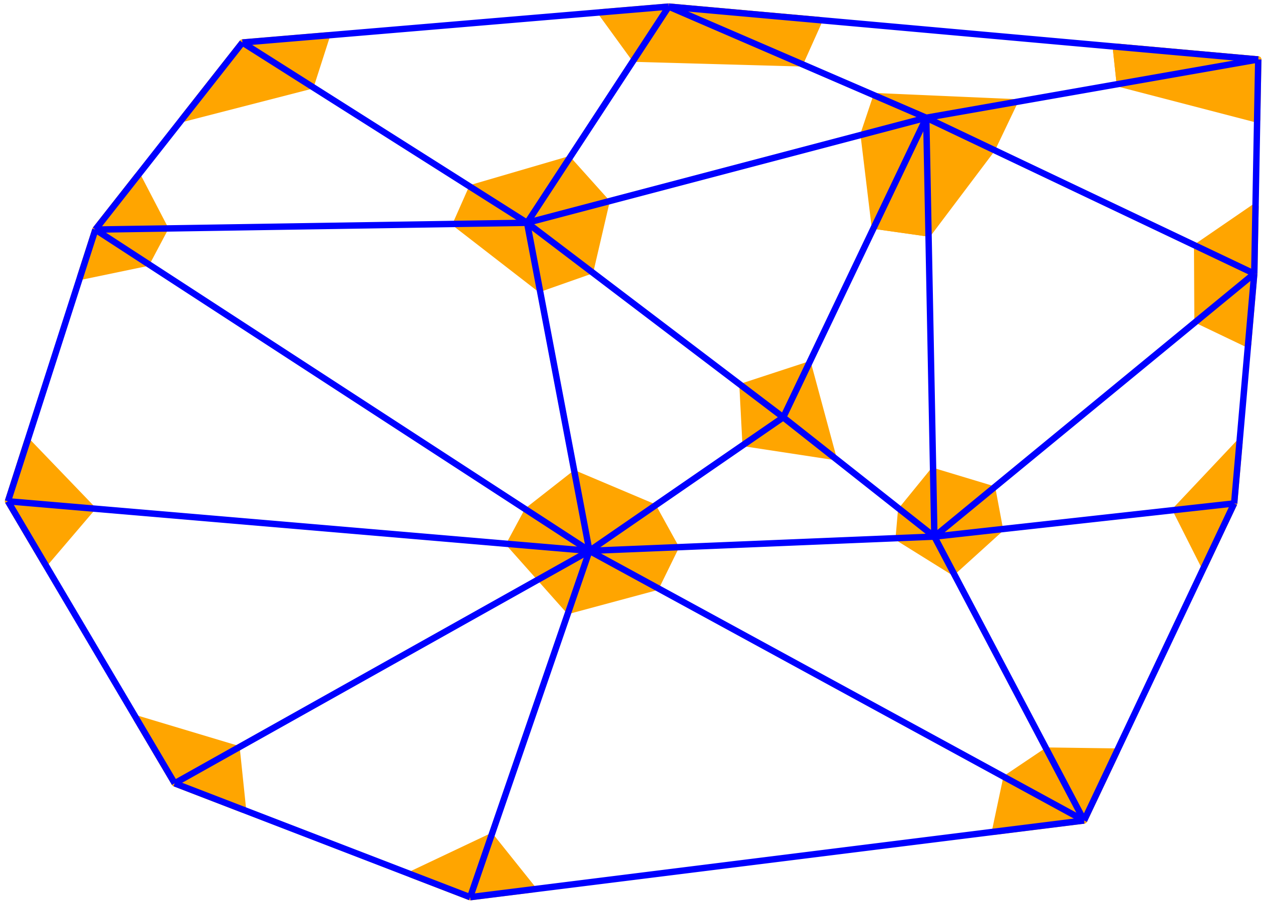


Delaunay Triangulation: max-min angle

Map: Triangulations $\longrightarrow \mathbb{R}^{6n-3k-4}$

$(\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_{6n-3k-4})$

smallest angle α_1
second smallest angle α_2
third smallest angle α_3

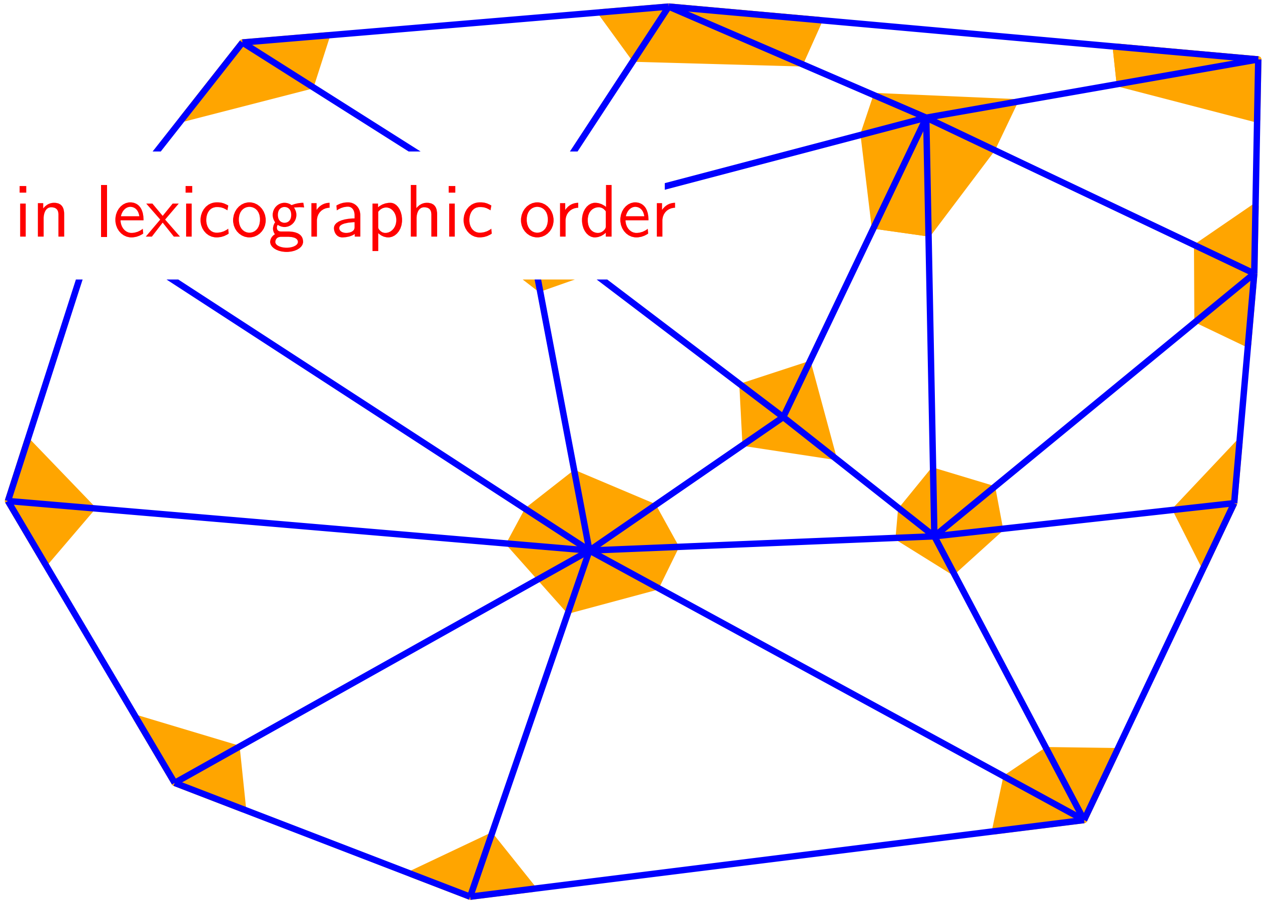


Delaunay Triangulation: max-min angle

Map: Triangulations $\longrightarrow \mathbb{R}^{6n-3k-4}$

$(\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_{6n-3k-4})$

smallest angle α_1
second smallest angle α_2
third smallest angle α_3



sort triangulations in lexicographic order

Delaunay Triangulation: max-min angle

Theorem:

Delaunay maximizes minimum angles (in lexicographic order)

Delaunay Triangulation: max-min angle

Theorem:

Delaunay maximizes minimum angles (in lexicographic order)

Proof:

Let T be the triangulation maximizing angles

Delaunay Triangulation: max-min angle

Theorem:

Delaunay maximizes minimum angles (in lexicographic order)

Proof:

Let T be the triangulation maximizing angles

$\Longrightarrow \forall$ convex quadrilateral (from 2 triangles $\in T$)

the diagonal maximizes smallest angle (in quad)

Delaunay Triangulation: max-min angle

Theorem:

Delaunay maximizes minimum angles (in lexicographic order)

Proof:

Let T be the triangulation maximizing angles

$\Longrightarrow \forall$ convex quadrilateral (from 2 triangles $\in T$)

the diagonal maximizes smallest angle (in quad)

$\Longrightarrow \forall$ edge, it is locally Delaunay

Delaunay Triangulation: max-min angle

Theorem:

Delaunay maximizes minimum angles (in lexicographic order)

Proof:

Let T be the triangulation maximizing angles

$\Longrightarrow \forall$ convex quadrilateral (from 2 triangles $\in T$)

the diagonal maximizes smallest angle (in quad)

$\Longrightarrow \forall$ edge, it is locally Delaunay

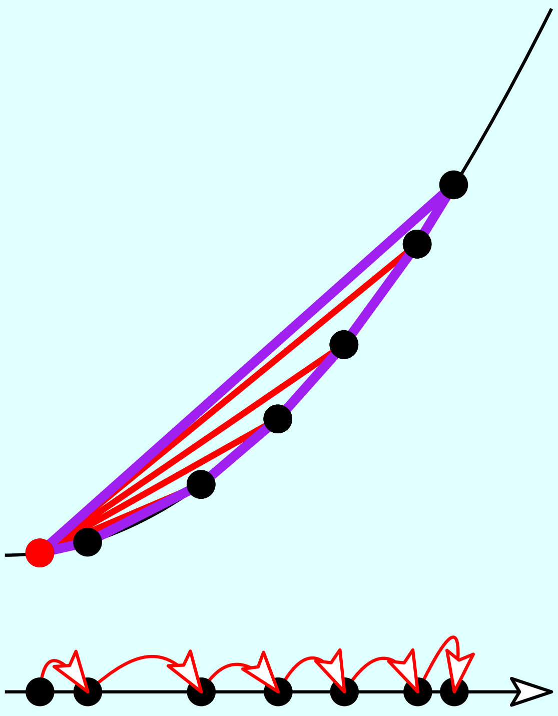
$\Longrightarrow T =$ Delaunay

Delaunay Triangulation: lower bound

Convex hull

Lower bound

A stupid algorithm for sorting numbers



$O(n)$ project on parabola

$f(n)$ compute convex hull

$O(n)$ find lowest point

$O(n)$ enumerate x coordinates in ccw CH order

Lower bound on sorting

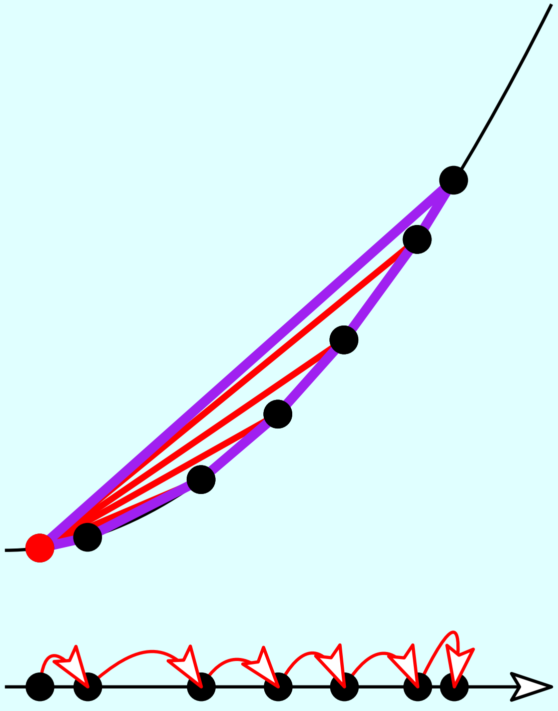
$\Longrightarrow f(n) + O(n) \geq \Omega(n \log n)$

Delaunay Triangulation: lower bound

Convex hull

Lower bound

A stupid algorithm for sorting numbers



$O(n)$ project on parabola

$f(n)$ compute ~~convex hull~~ Delaunay

$O(n)$ find lowest point

$O(n)$ enumerate x coordinates in ccw CH order

Lower bound on sorting

$$\Longrightarrow f(n) + O(n) \geq \Omega(n \log n)$$

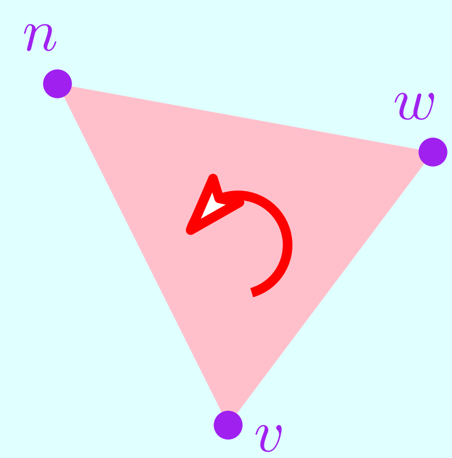
Delaunay Triangulation: incircle predicate

Convex hull

Orientation predicate

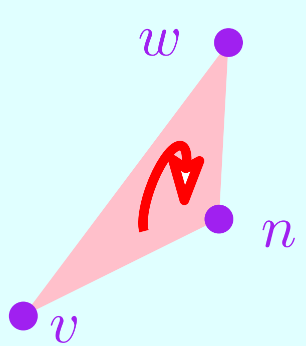
vwn + ?

$$\begin{vmatrix} x_w - x_v & x_n - x_v \\ y_w - y_v & y_n - y_v \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 \\ x_v & x_w & x_n \\ y_v & y_w & y_n \end{vmatrix} > 0$$



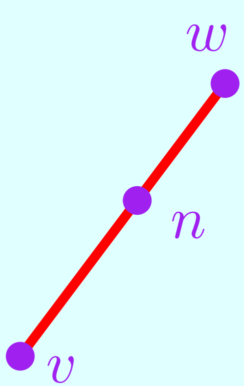
vwn - ?

$$\begin{vmatrix} 1 & 1 & 1 \\ x_v & x_w & x_n \\ y_v & y_w & y_n \end{vmatrix} < 0$$

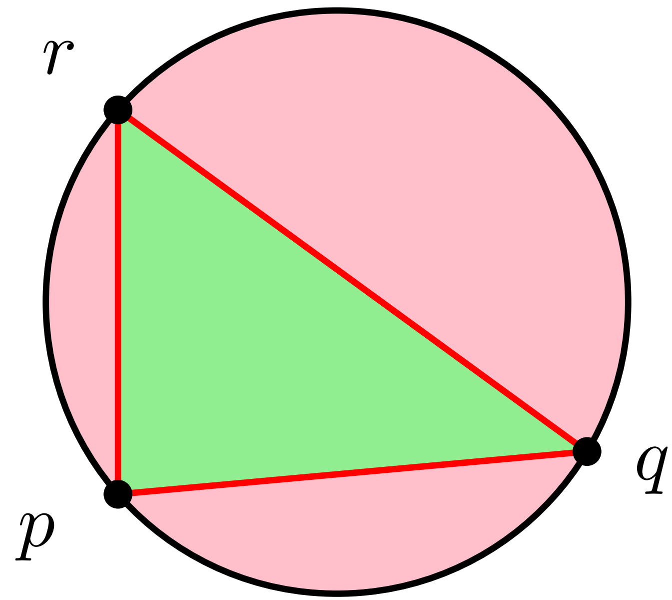


vwn 0 ?

$$\begin{vmatrix} 1 & 1 & 1 \\ x_v & x_w & x_n \\ y_v & y_w & y_n \end{vmatrix} = 0$$



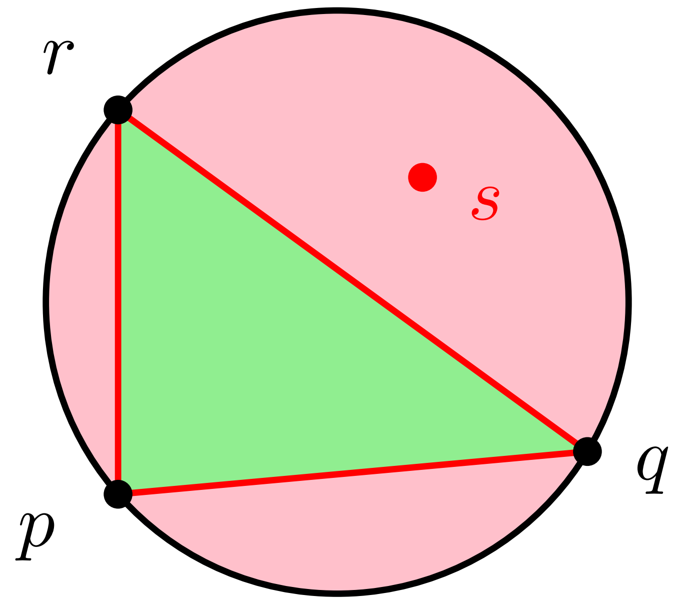
Delaunay Triangulation: incircle predicate



pqr ccw triangle

query s

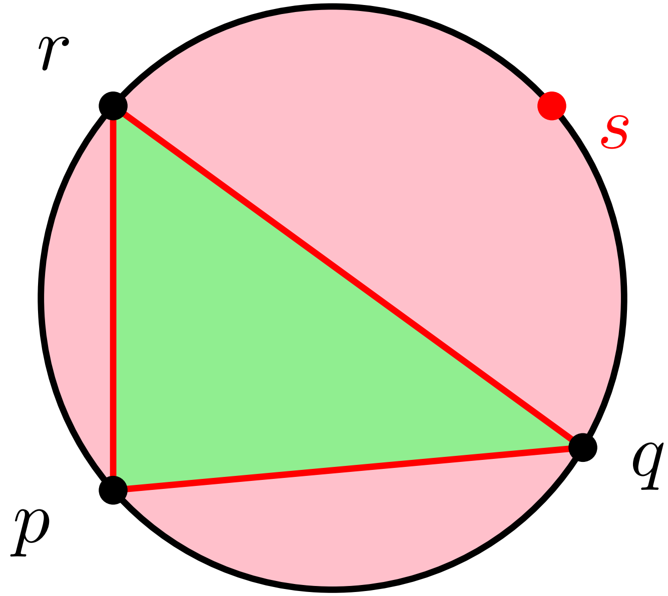
Delaunay Triangulation: incircle predicate



pqr ccw triangle

query s inside circumcircle

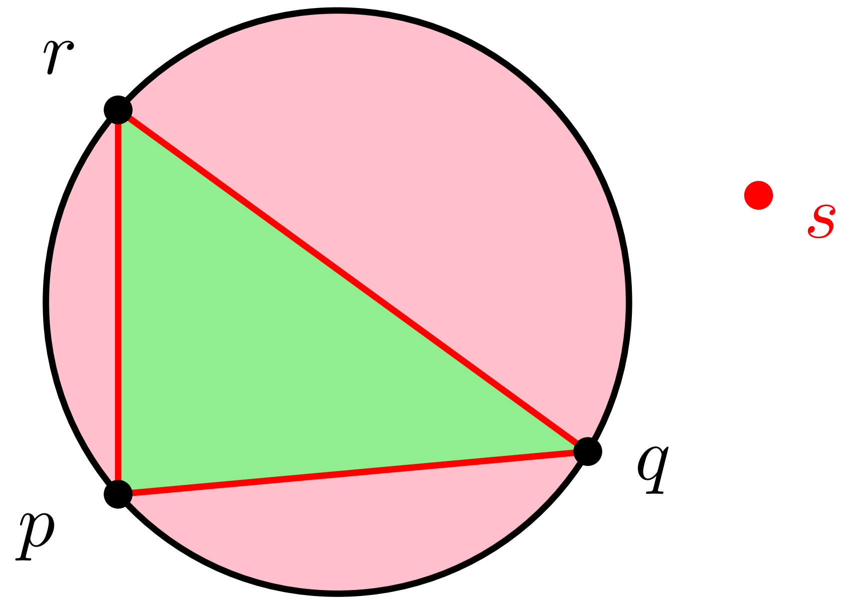
Delaunay Triangulation: incircle predicate



pqr ccw triangle

query s cocircular

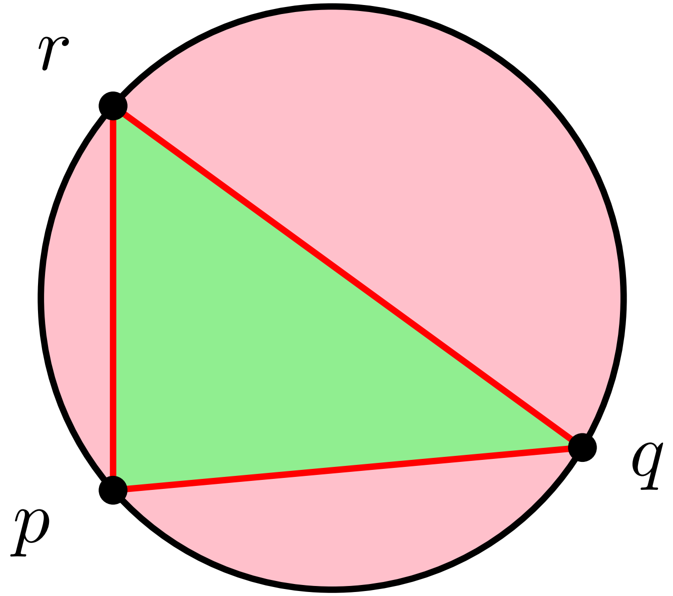
Delaunay Triangulation: incircle predicate



pqr ccw triangle

query s outside circumcircle

Delaunay Triangulation: incircle predicate

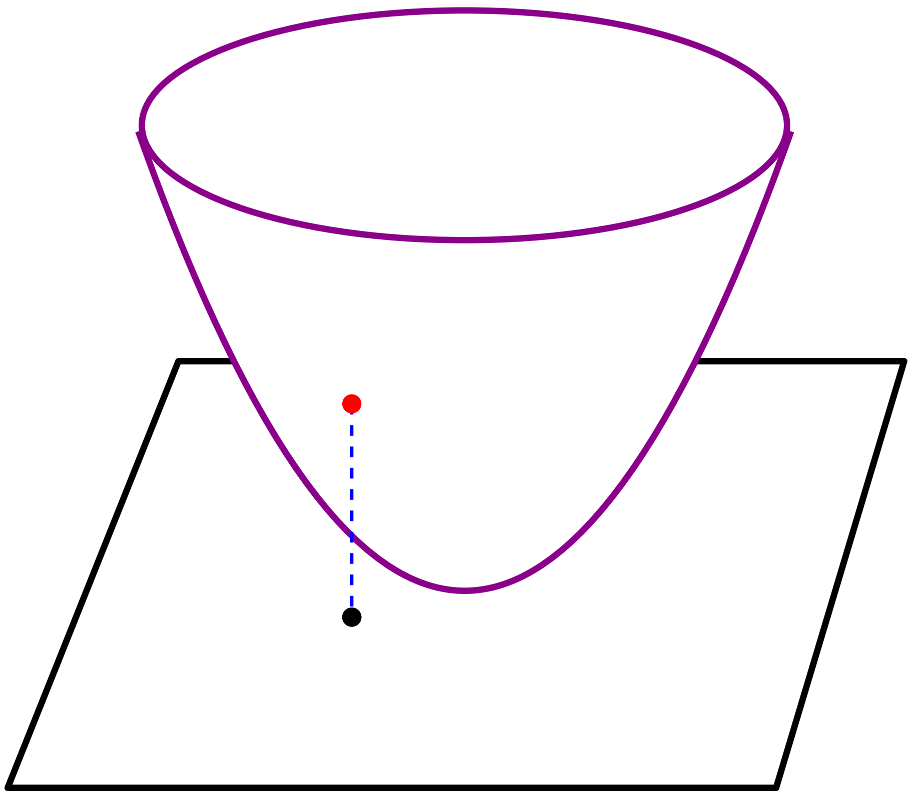


pqr ccw triangle

query s

Delaunay Triangulation: incircle predicate

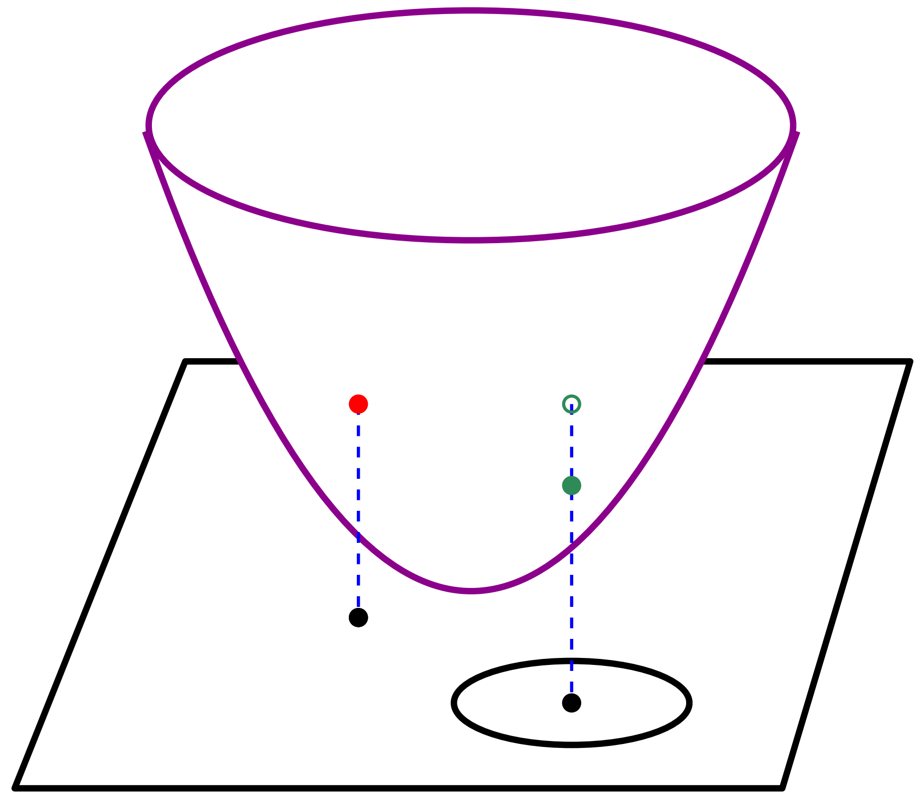
Space of circles



$$p = (x, y) \rightsquigarrow p^\star = (x, y, x^2 + y^2)$$

Delaunay Triangulation: incircle predicate

Space of circles



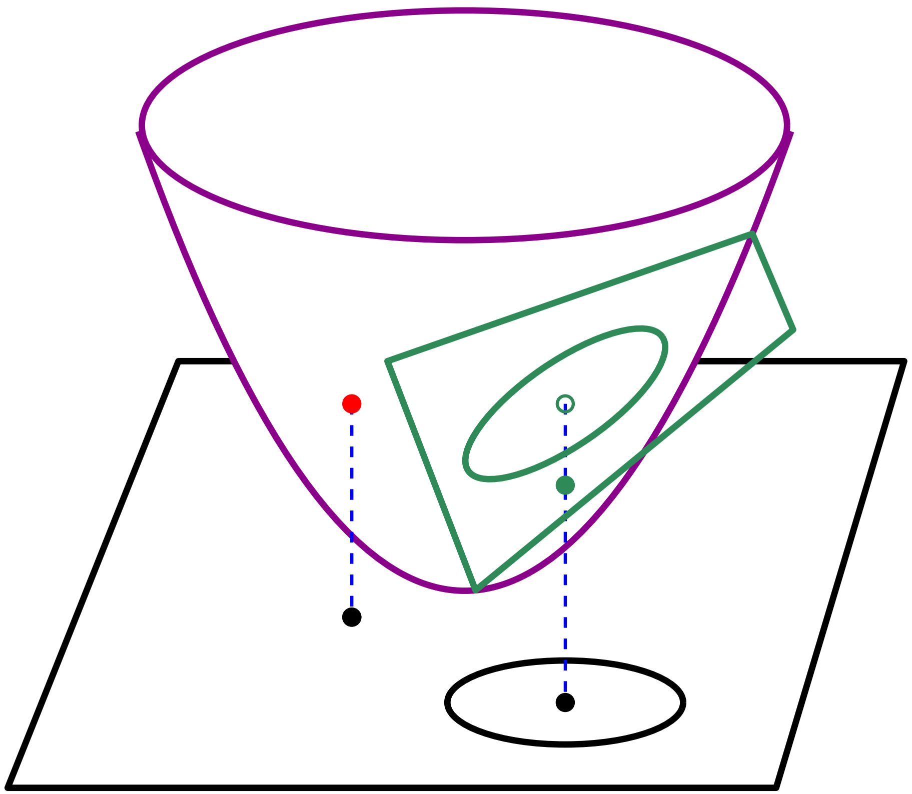
$$p = (x, y) \rightsquigarrow p^\star = (x, y, x^2 + y^2)$$

$$C : x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$$

$$\rightsquigarrow C^\star = (a, b, a^2 + b^2 - r^2)$$

Delaunay Triangulation: incircle predicate

Space of circles



$$p = (x, y) \rightsquigarrow p^\star = (x, y, x^2 + y^2)$$

$$C : x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$$

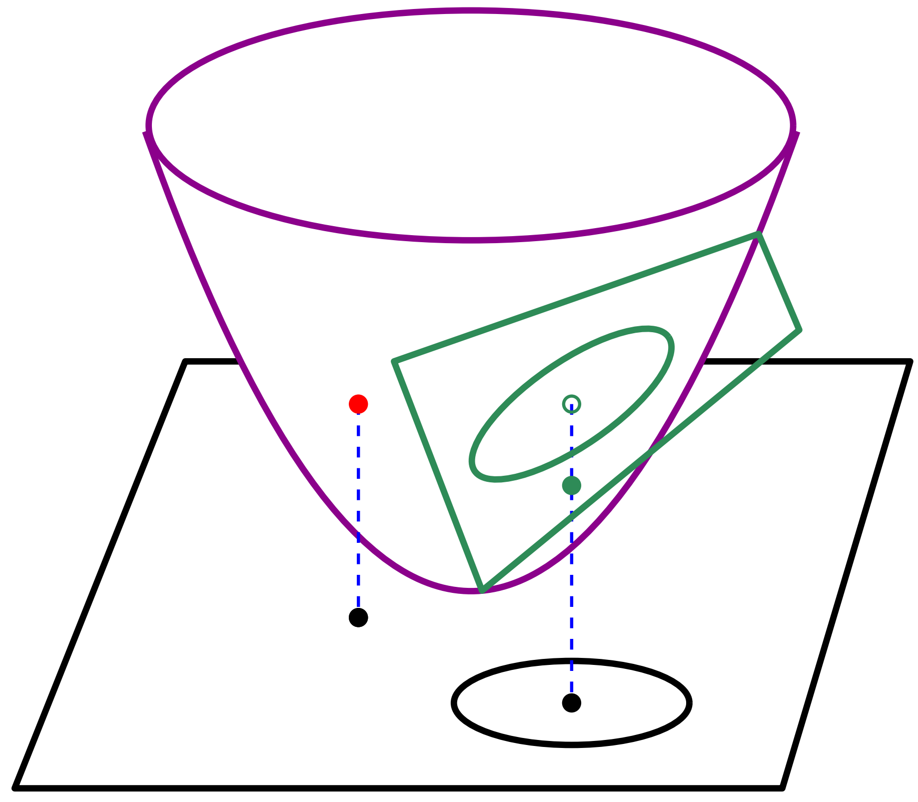
$$\rightsquigarrow C^\star = (a, b, a^2 + b^2 - r^2)$$

$$\rightsquigarrow C^\dagger : z - 2ax - 2by + a^2 + b^2 - r^2 = 0$$

Delaunay Triangulation: incircle predicate

Space of circles

$p \in C \iff p^\star \in C^\dagger$



$p = (x, y) \rightsquigarrow p^\star = (x, y, x^2 + y^2)$

$C : x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$

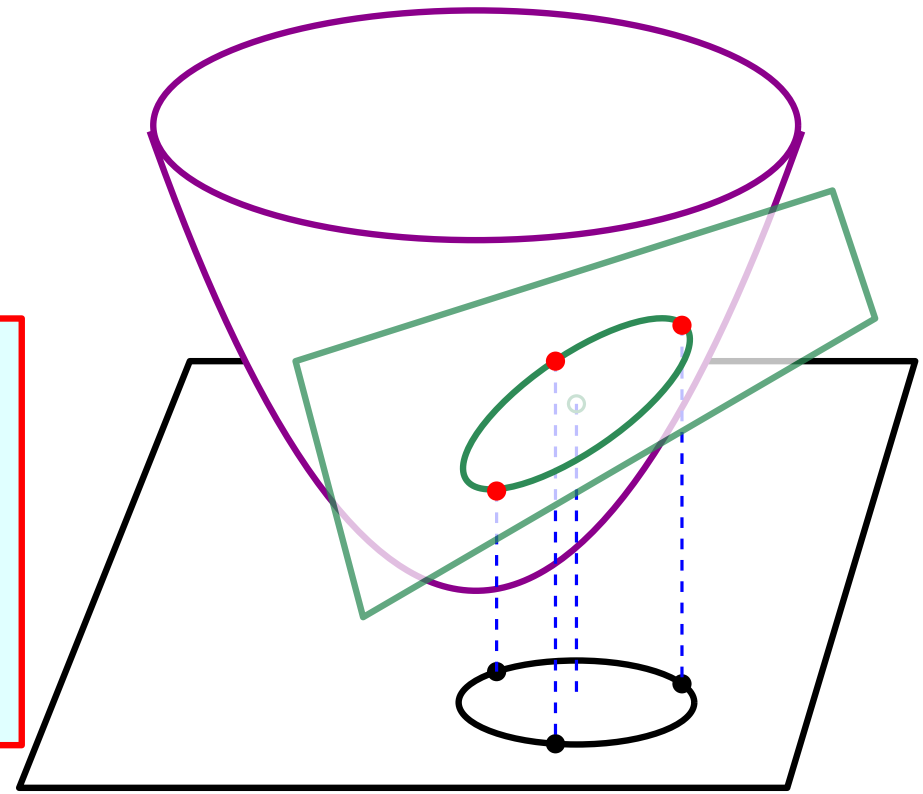
$\rightsquigarrow C^\star = (a, b, a^2 + b^2 - r^2)$

$\rightsquigarrow C^\dagger : z - 2ax - 2by + a^2 + b^2 - r^2 = 0$

Delaunay Triangulation: incircle predicate

Space of circles

$p \in C \iff p^\star \in C^\dagger$



circle through pqr

$\rightsquigarrow$ plane through $p^\star q^\star r^\star$

$p = (x, y) \rightsquigarrow p^\star = (x, y, x^2 + y^2)$

$C : x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$

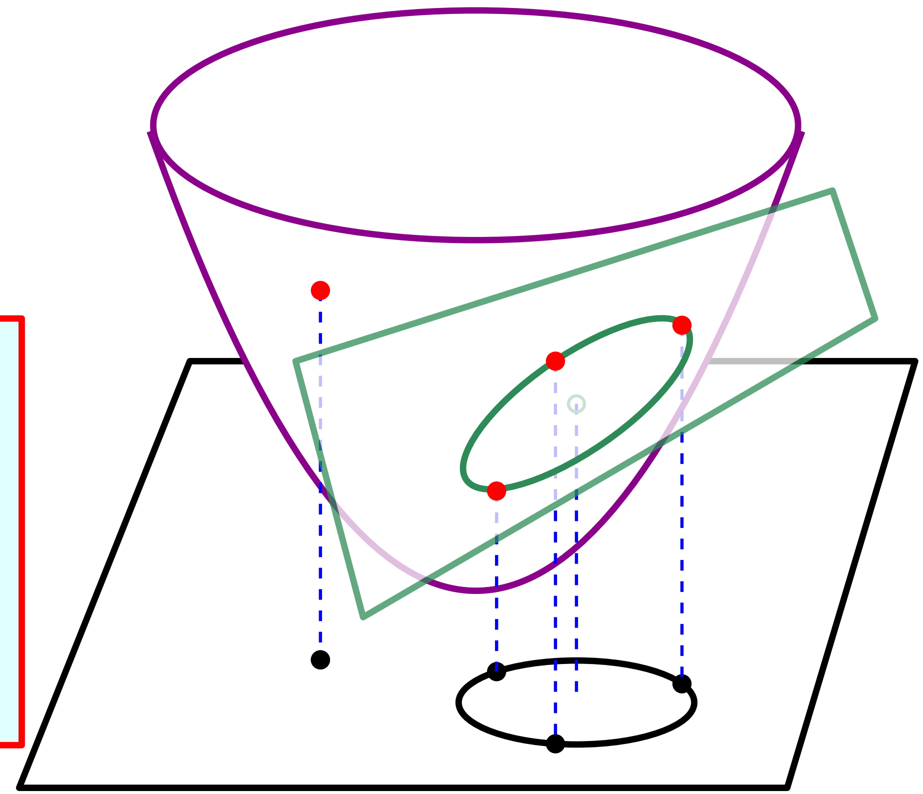
$\rightsquigarrow C^\star = (a, b, a^2 + b^2 - r^2)$

$\rightsquigarrow C^\dagger : z - 2ax - 2by + a^2 + b^2 - r^2 = 0$

Delaunay Triangulation: incircle predicate

Space of circles

$p \in C \iff p^\star \in C^\dagger$



s inside/outside of
circle through pqr
$\rightsquigarrow$ plane through $p^\star q^\star r^\star$
above/below $s^\star$

$p = (x, y) \rightsquigarrow p^\star = (x, y, x^2 + y^2)$

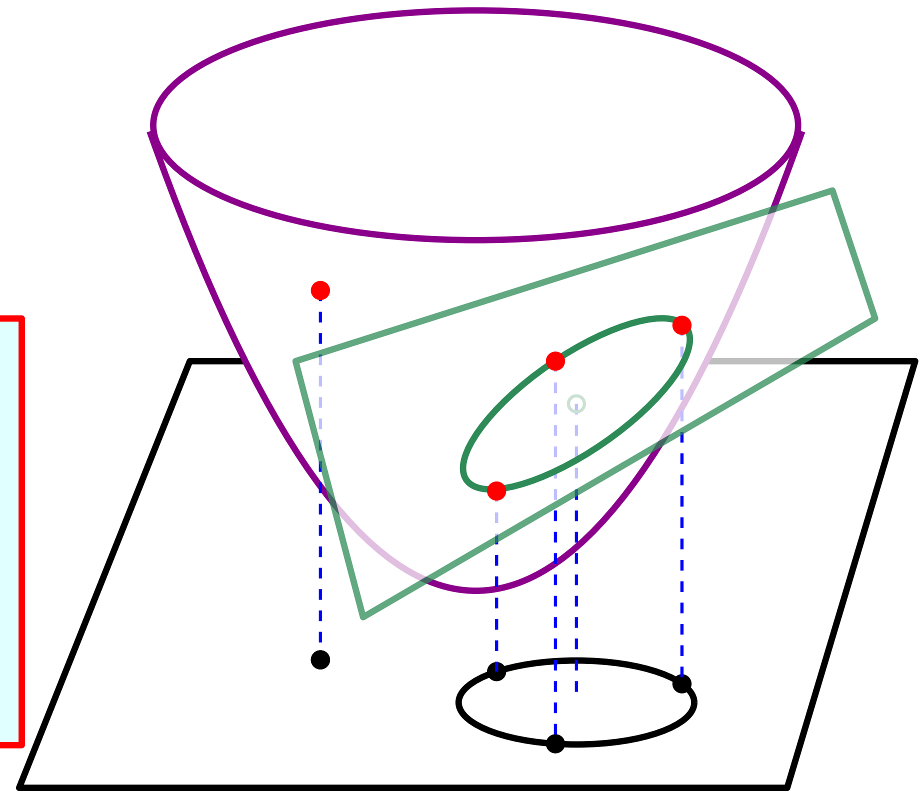
$C : x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$

$\rightsquigarrow C^\star = (a, b, a^2 + b^2 - r^2)$

$\rightsquigarrow C^\dagger : z - 2ax - 2by + a^2 + b^2 - r^2 = 0$

Delaunay Triangulation: incircle predicate

Space of circles



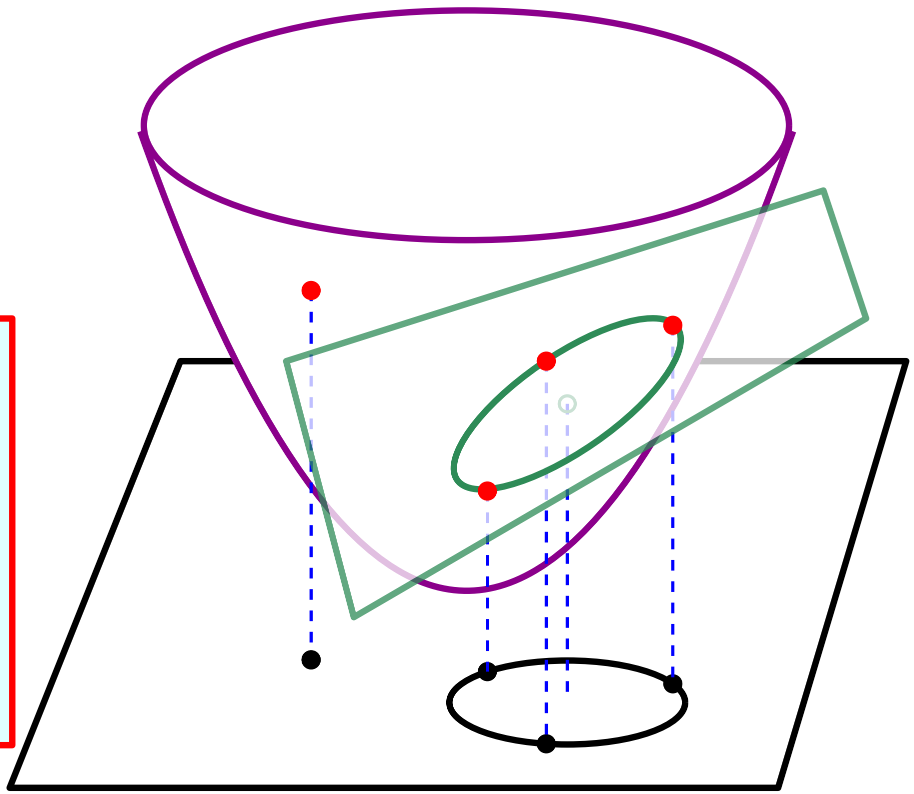
s inside/outside of circle through pqr
$\rightsquigarrow$ plane through $p^\star q^\star r^\star$ above/below $s^\star$

incircle predicate

$\rightsquigarrow$ 3D orientation predicate

Delaunay Triangulation: incircle predicate

Space of circles



s inside/outside of circle through pqr

$\leadsto$ plane through $p^\star q^\star r^\star$ above/below $s^\star$

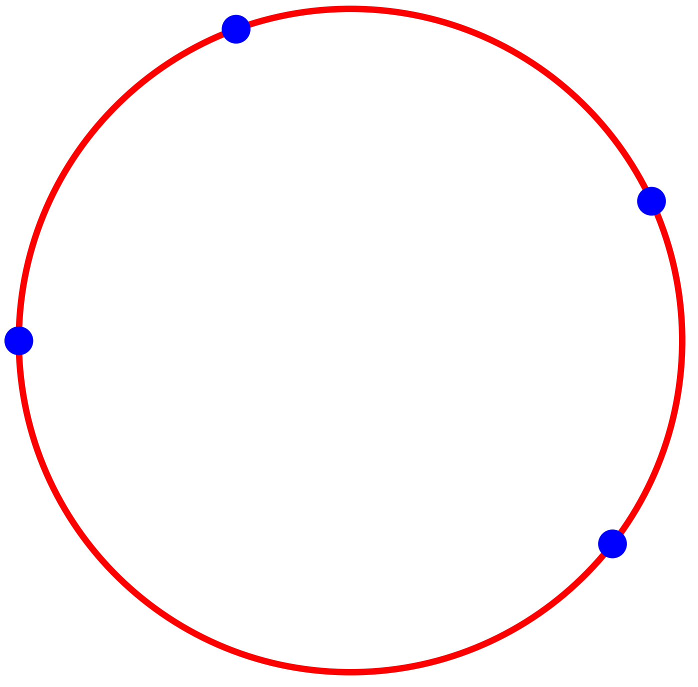
incircle predicate

$\leadsto$ 3D orientation predicate

$$\operatorname{sign}\begin{vmatrix} 1 & 1 & 1 & 1 \\ x_p & x_q & x_r & x_s \\ y_p & y_q & y_r & y_s \\ x_p^2+y_p^2 & x_q^2+y_q^2 & x_r^2+y_r^2 & x_s^2+y_s^2 \end{vmatrix}$$

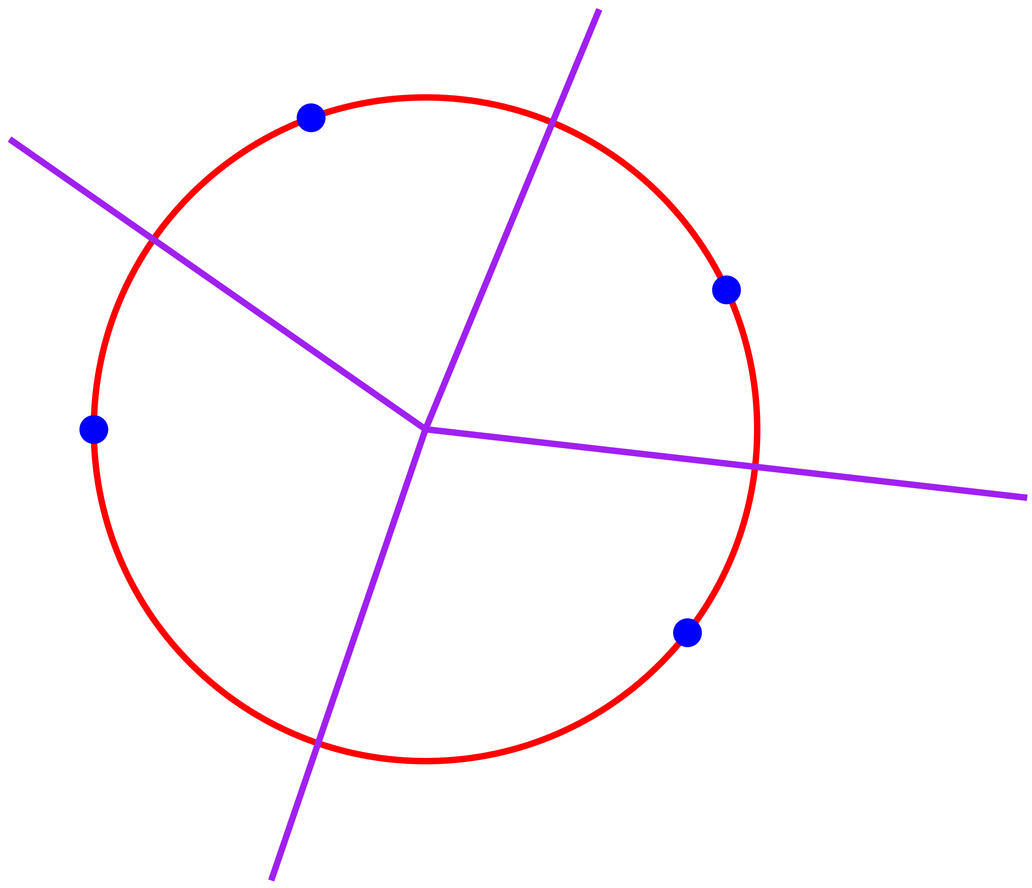
Delaunay Triangulation: incircle predicate

Degeneracies



Delaunay Triangulation: incircle predicate

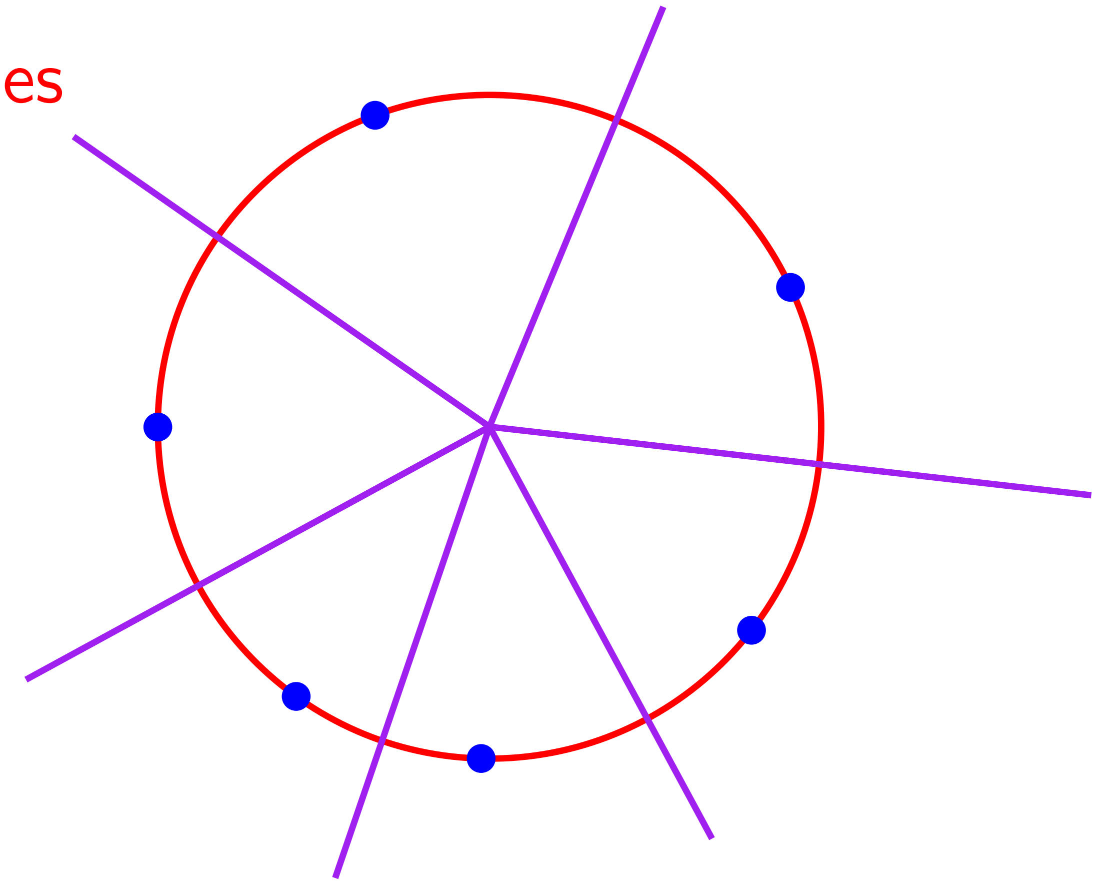
Degeneracies



Degree 4 vertex in Voronoi diagram

Delaunay Triangulation: incircle predicate

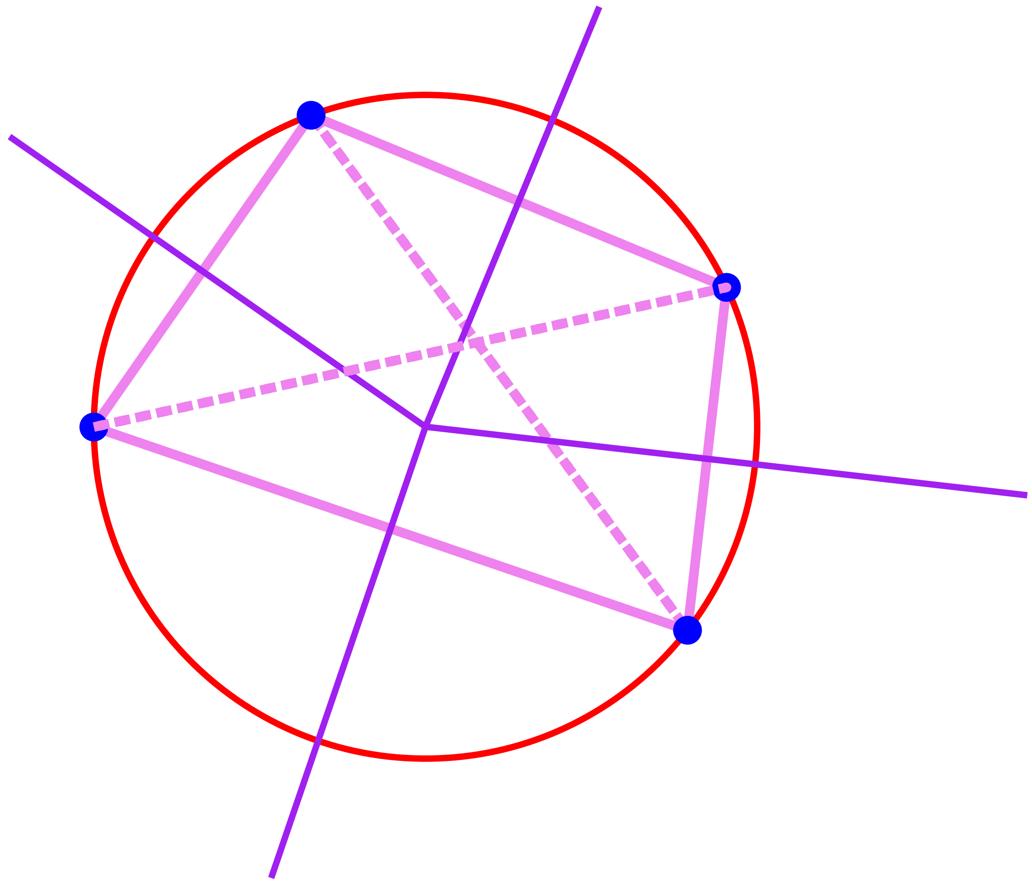
Degeneracies



Degree ~~4~~ d vertex in Voronoi diagram

Delaunay Triangulation: incircle predicate

Degeneracies

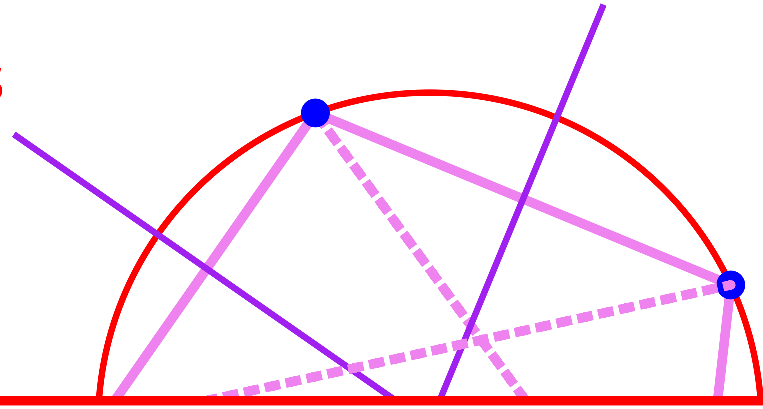


Degree 4 vertex in Voronoi diagram

Delaunay quad ? random diagonal ?

Delaunay Triangulation: incircle predicate

Degeneracies



Teaser robustness lecture

assume no degeneracies for a while

Degree 4 vertex in Voronoi diagram

Delaunay quad ? random diagonal ?

Delaunay Triangulation:

Teaser CGAL lecture

Data structure for (Delaunay) triangulation ?

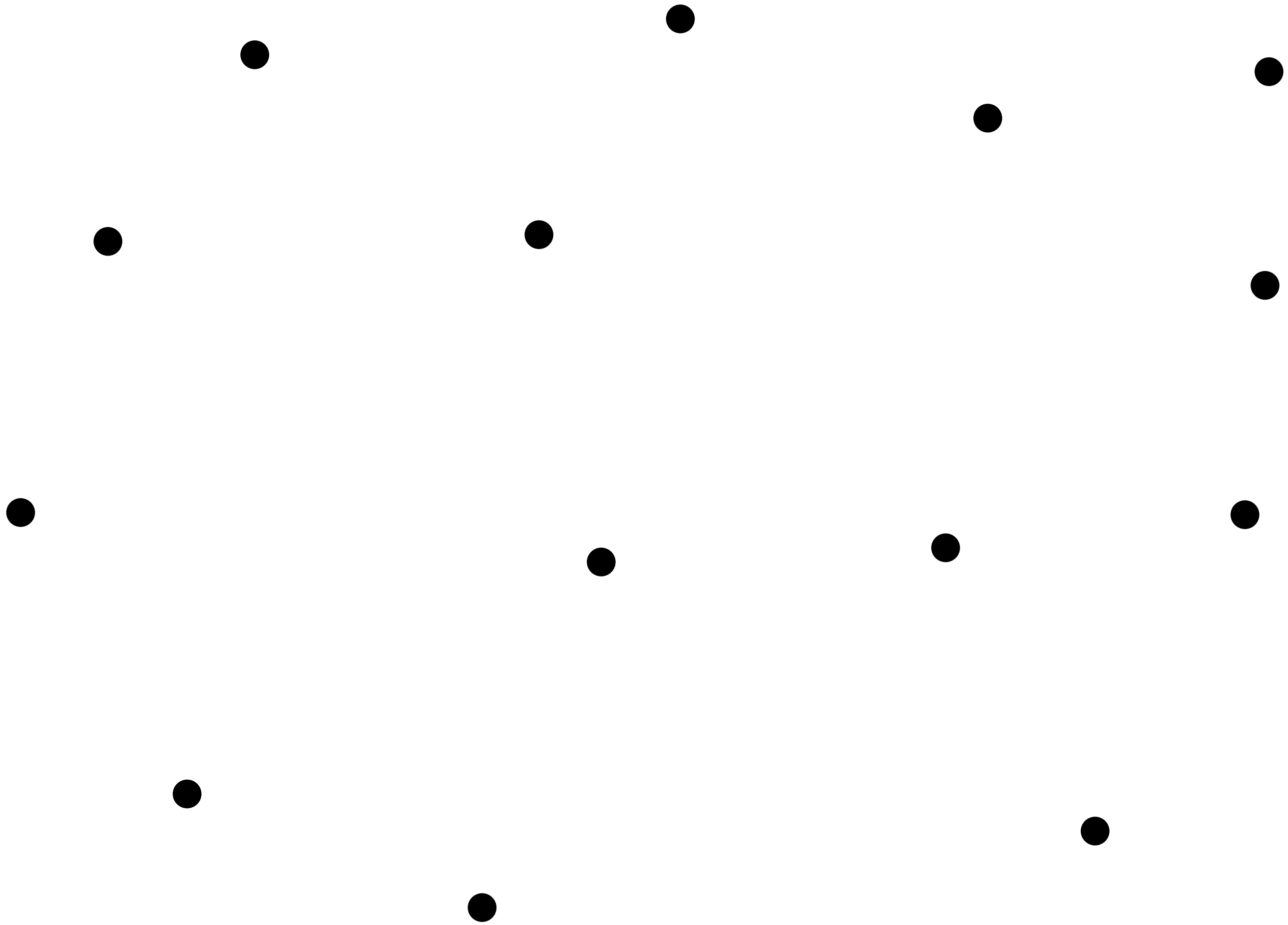
Representing incidences

Representing hull boundary

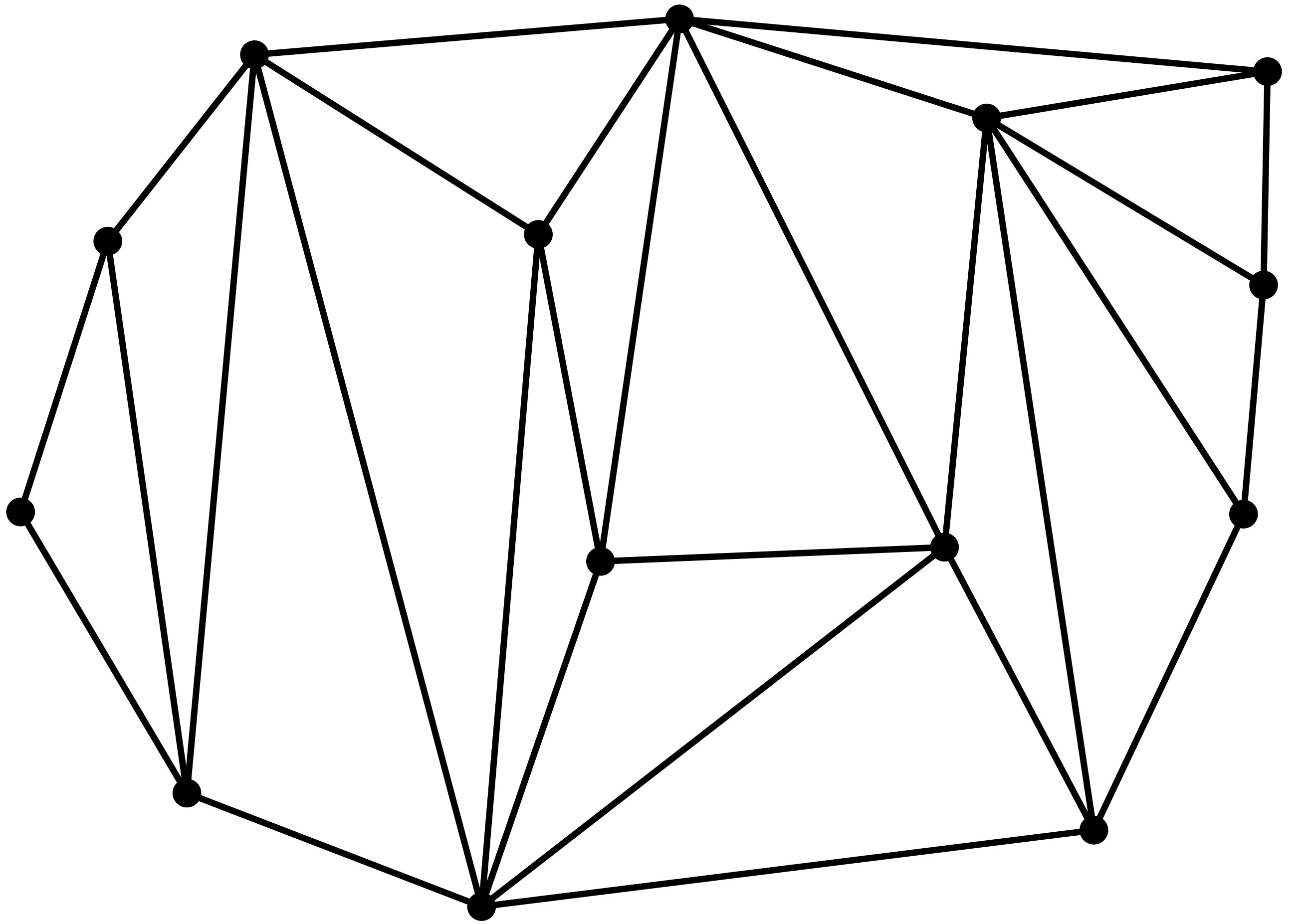
Representing user's data put colors in triangles

...

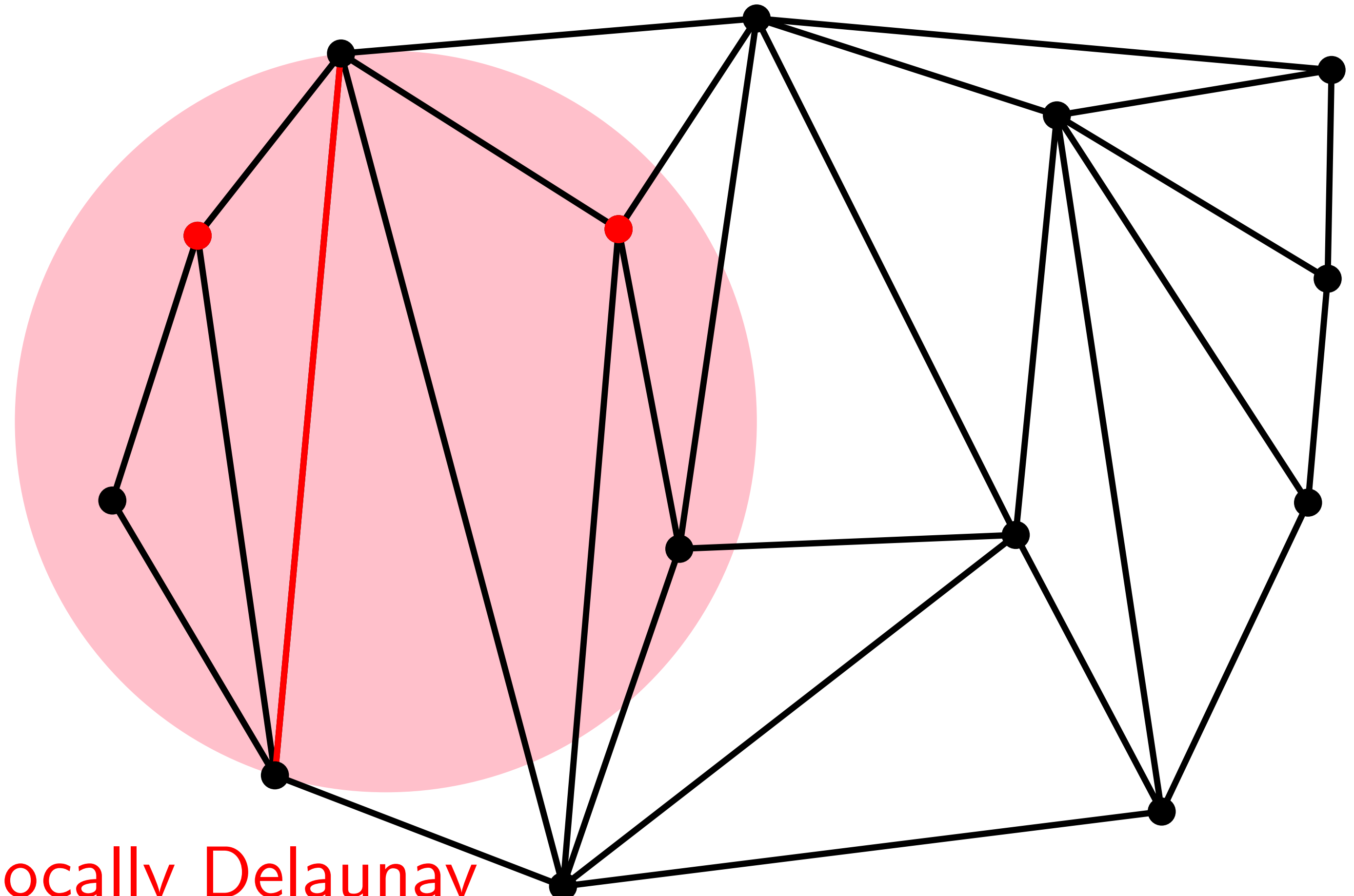
Delaunay Triangulation: Diagonal flipping



Delaunay Triangulation: Diagonal flipping

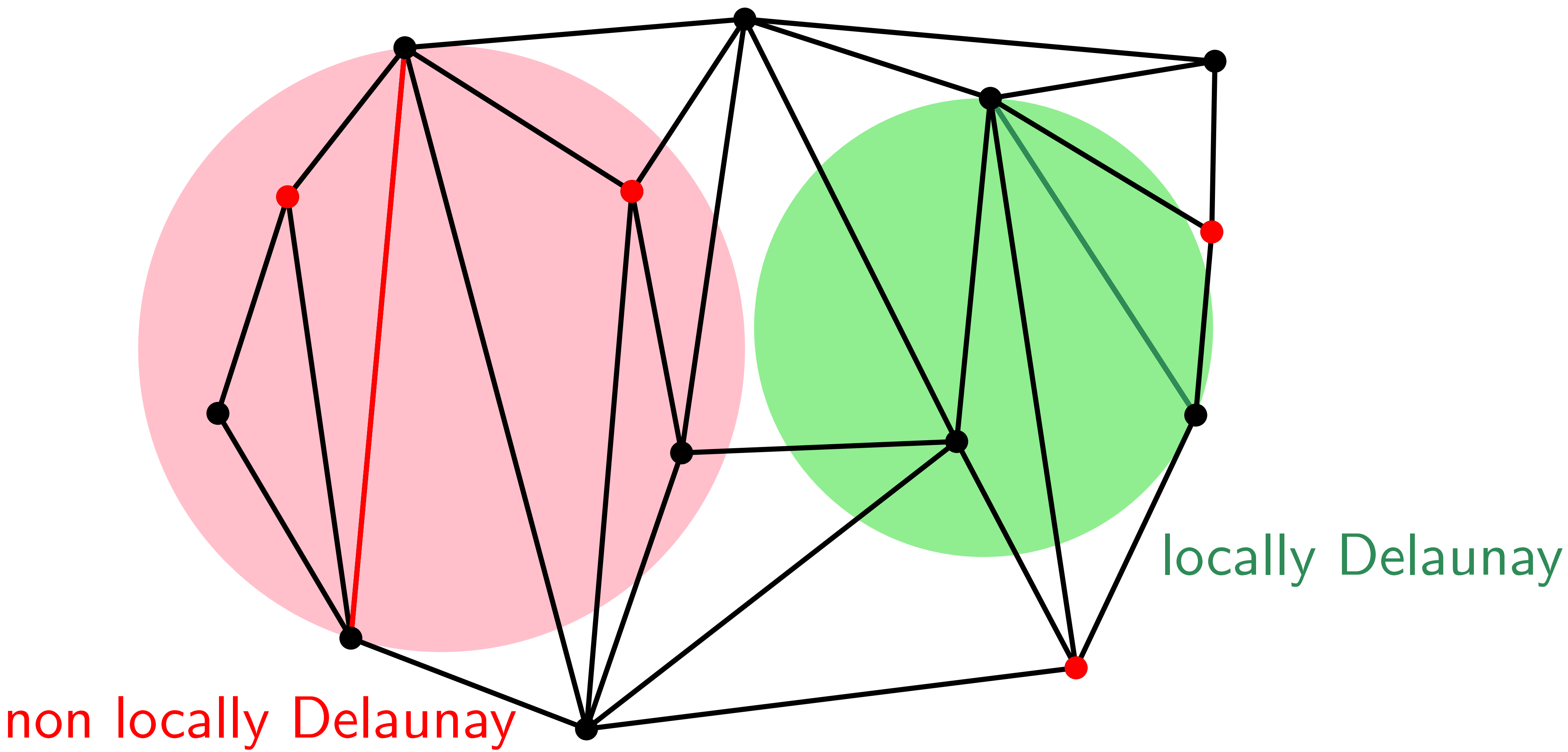


Delaunay Triangulation: Diagonal flipping

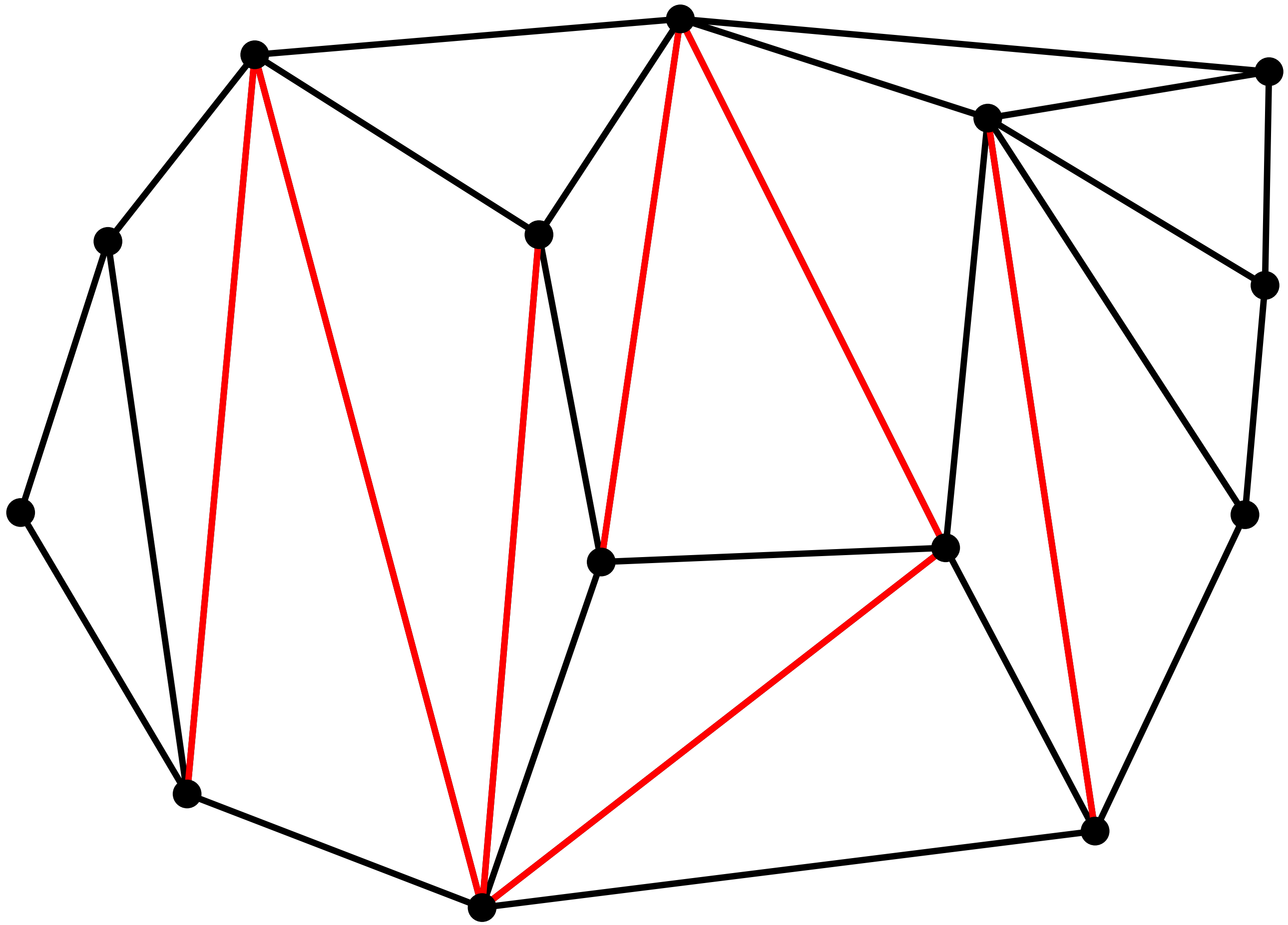


non locally Delaunay

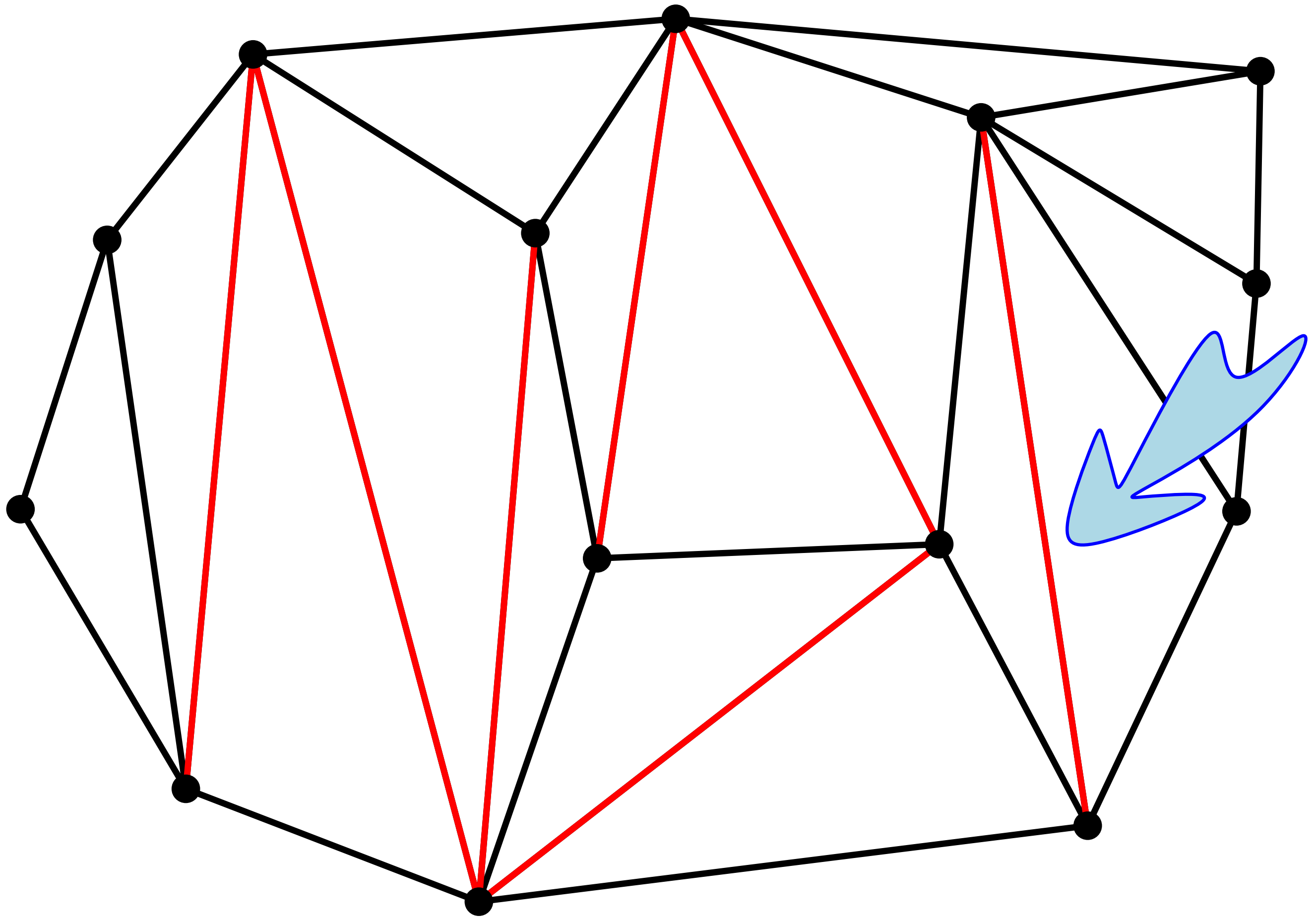
Delaunay Triangulation: Diagonal flipping



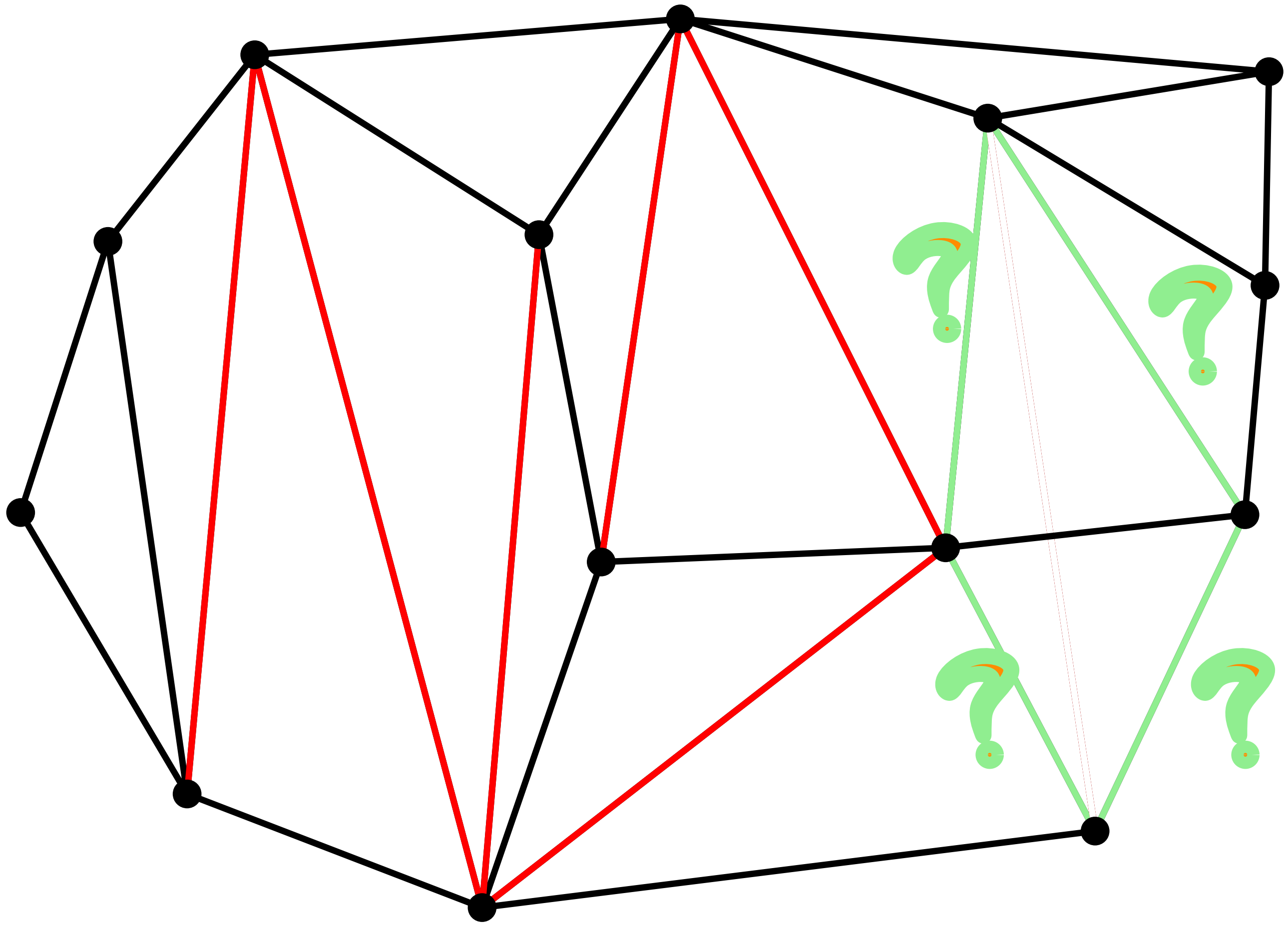
Delaunay Triangulation: Diagonal flipping



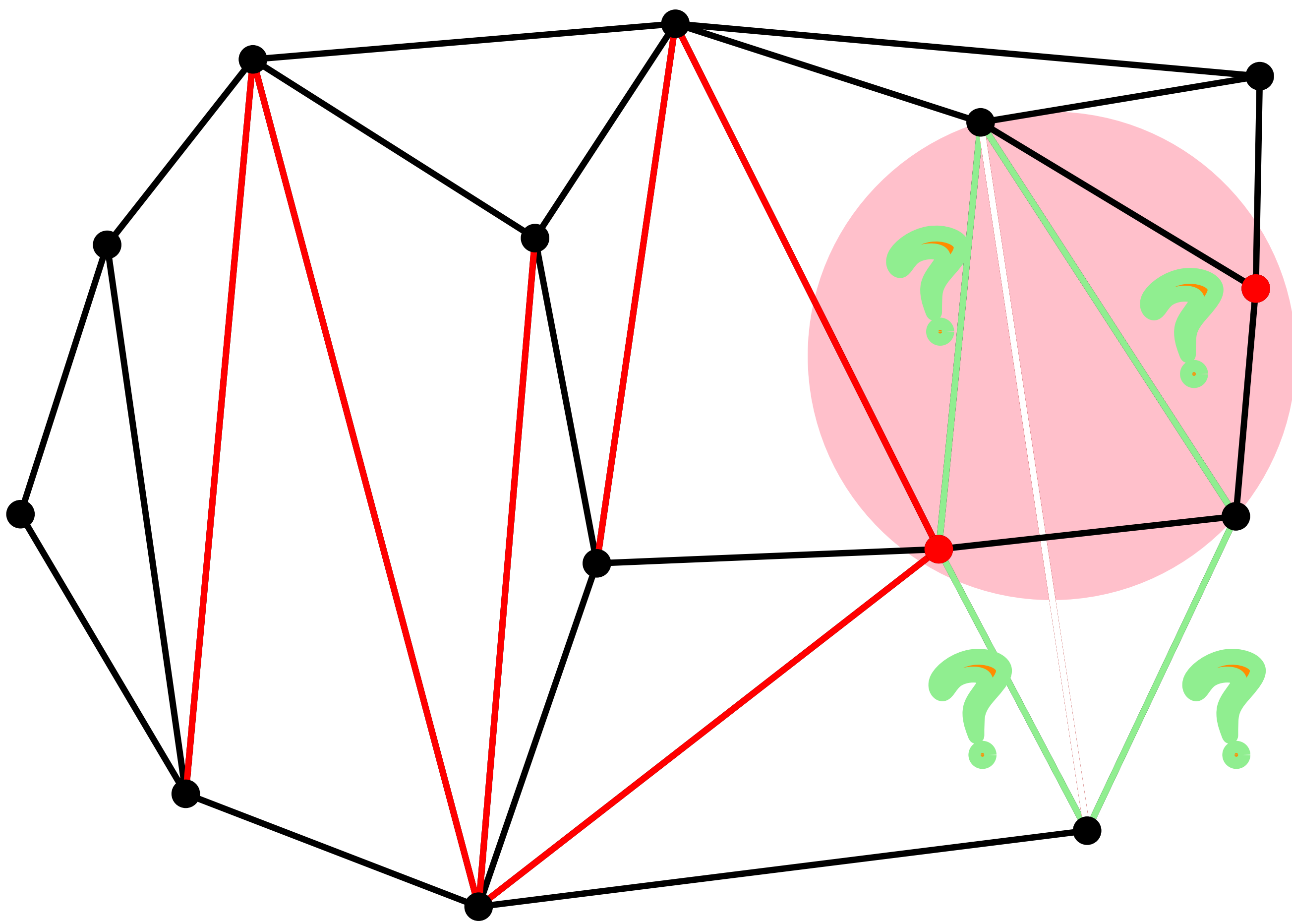
Delaunay Triangulation: Diagonal flipping



Delaunay Triangulation: Diagonal flipping

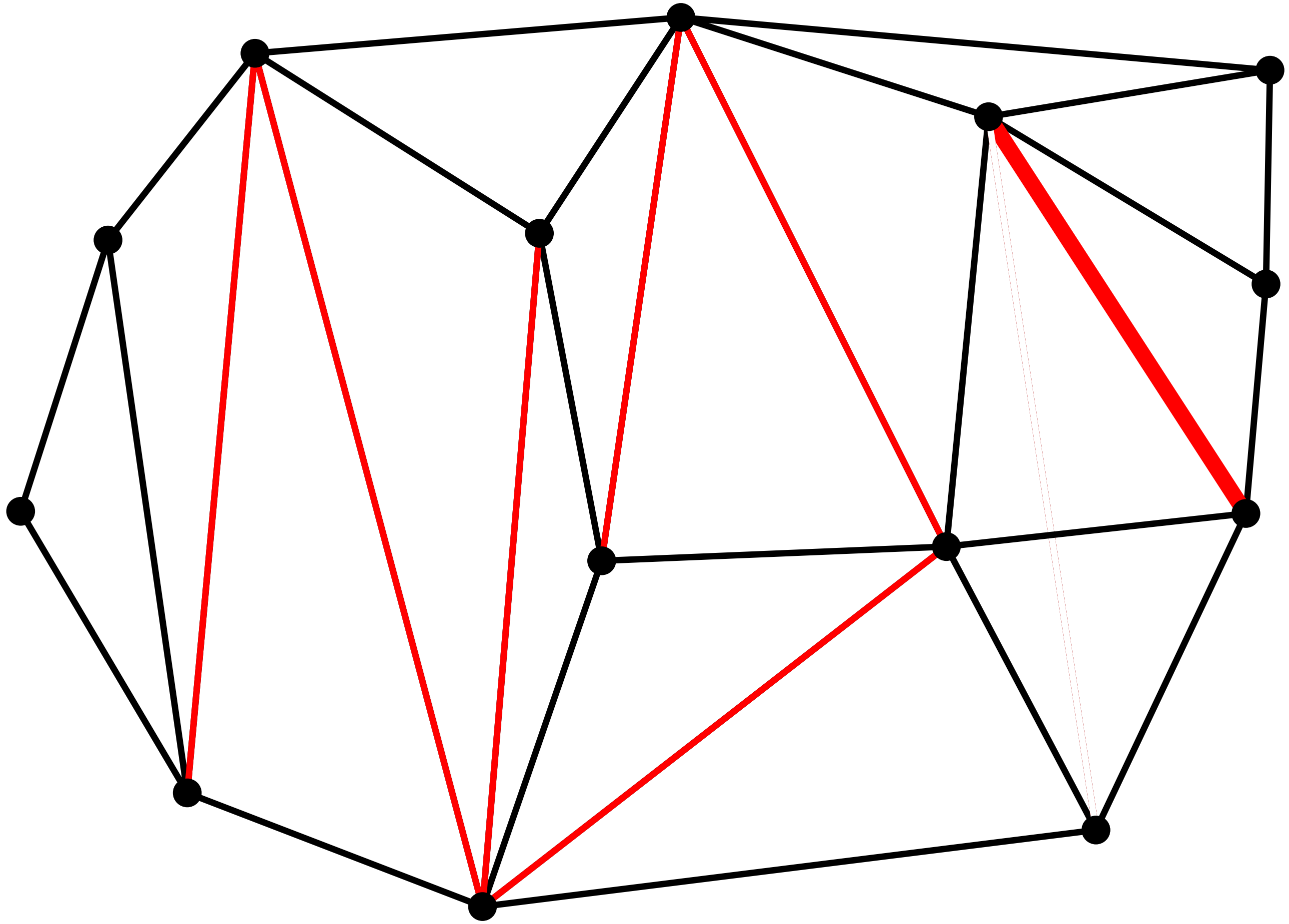


Delaunay Triangulation: Diagonal flipping

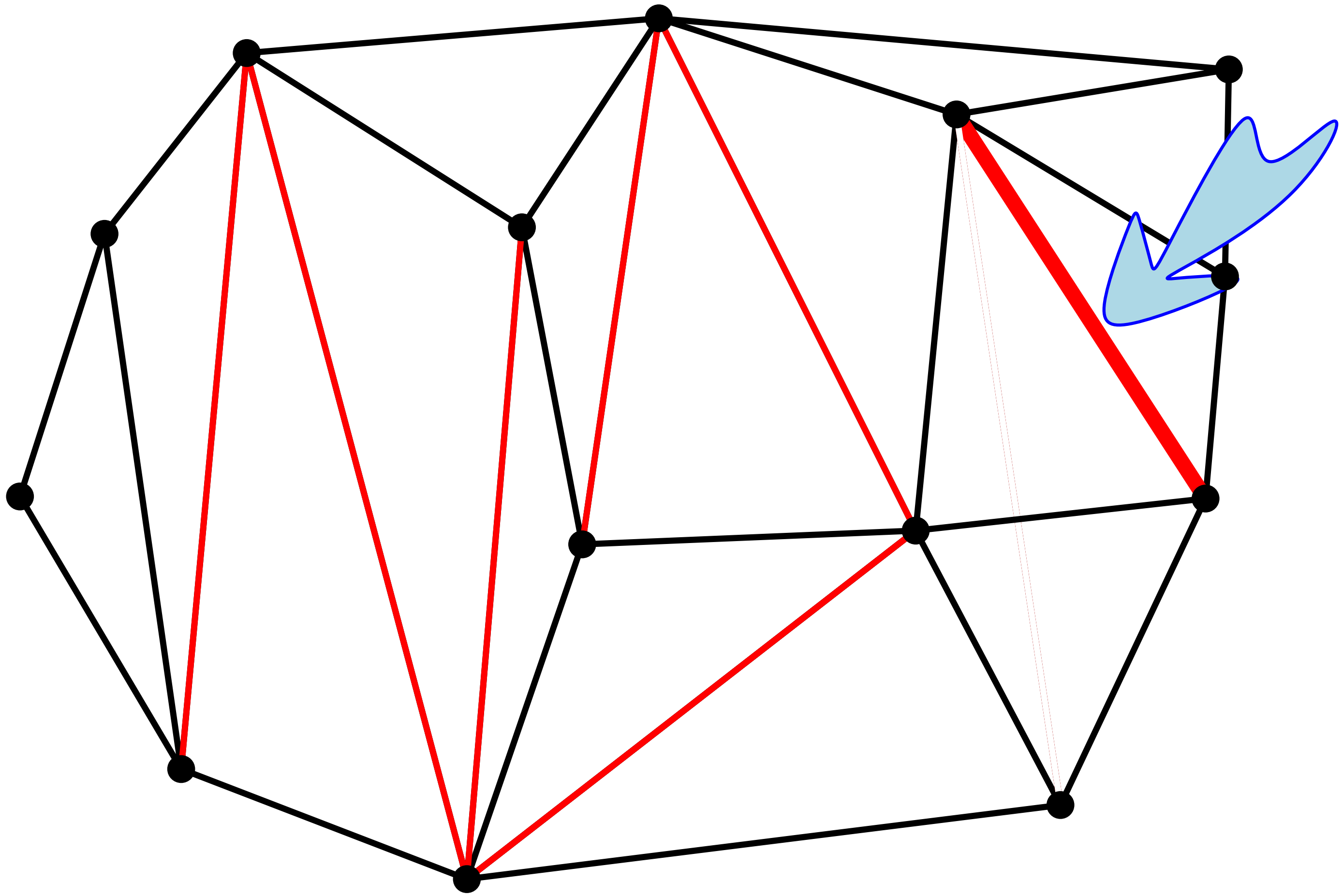


check edges of quadrilateral

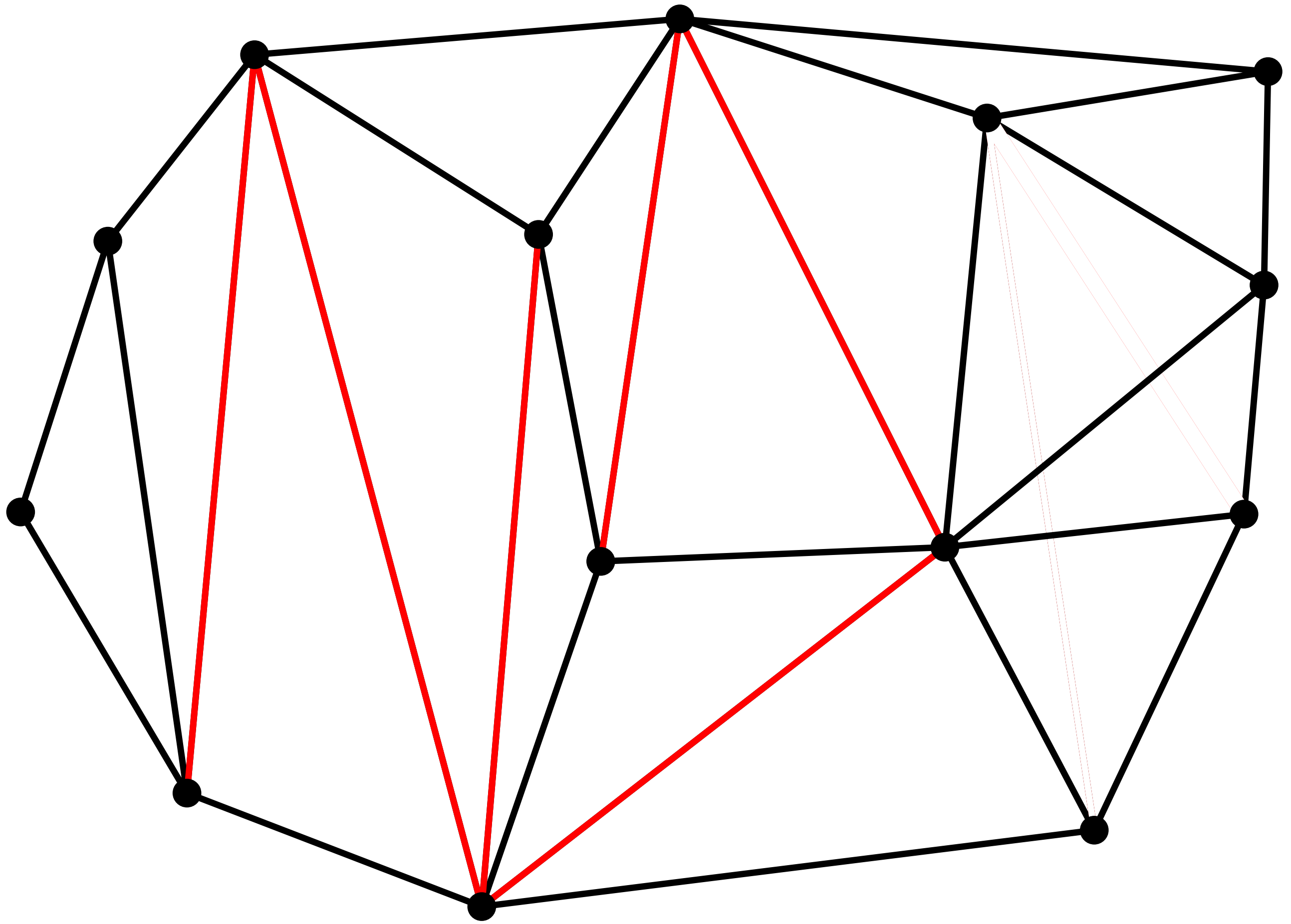
Delaunay Triangulation: Diagonal flipping



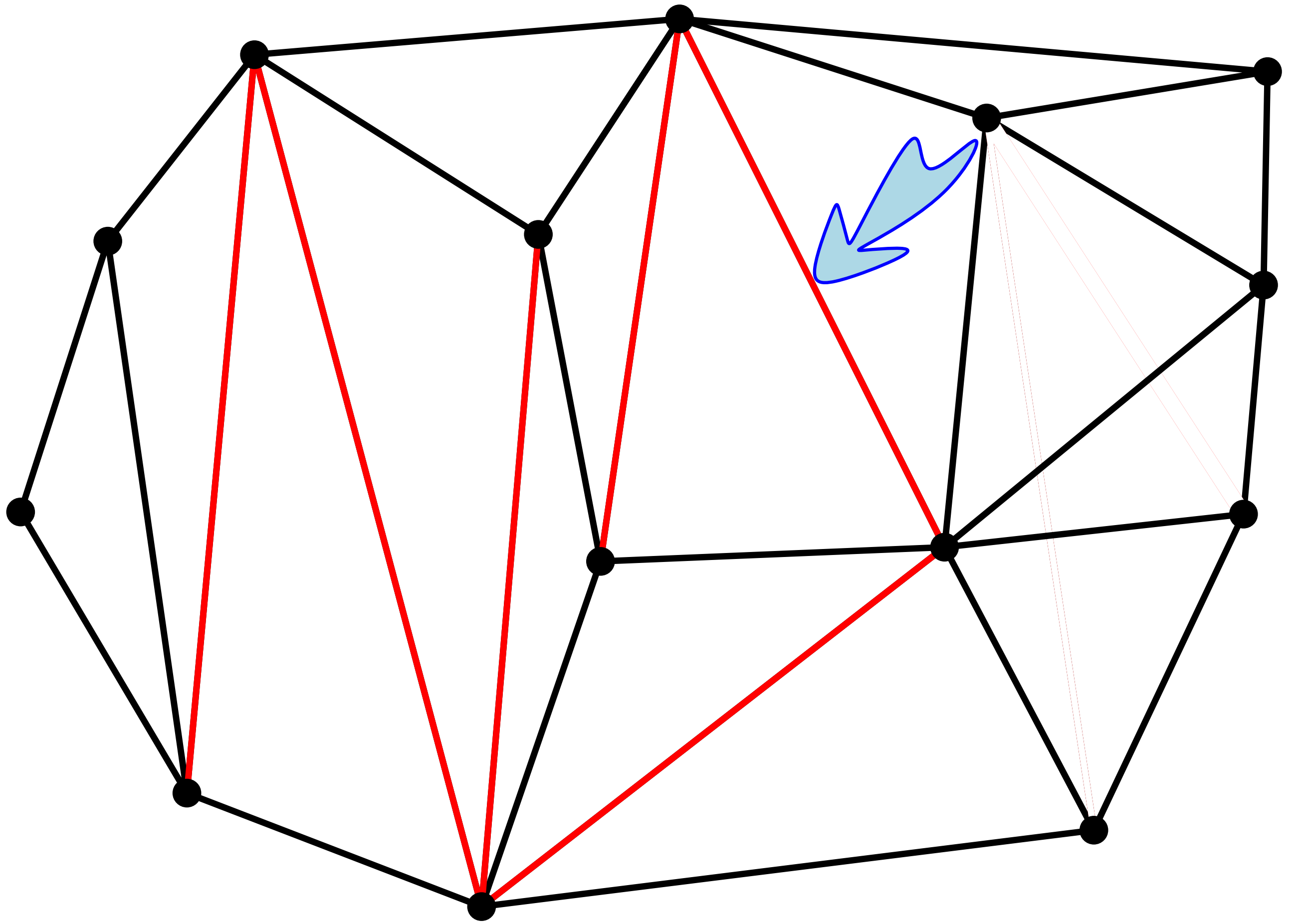
Delaunay Triangulation: Diagonal flipping



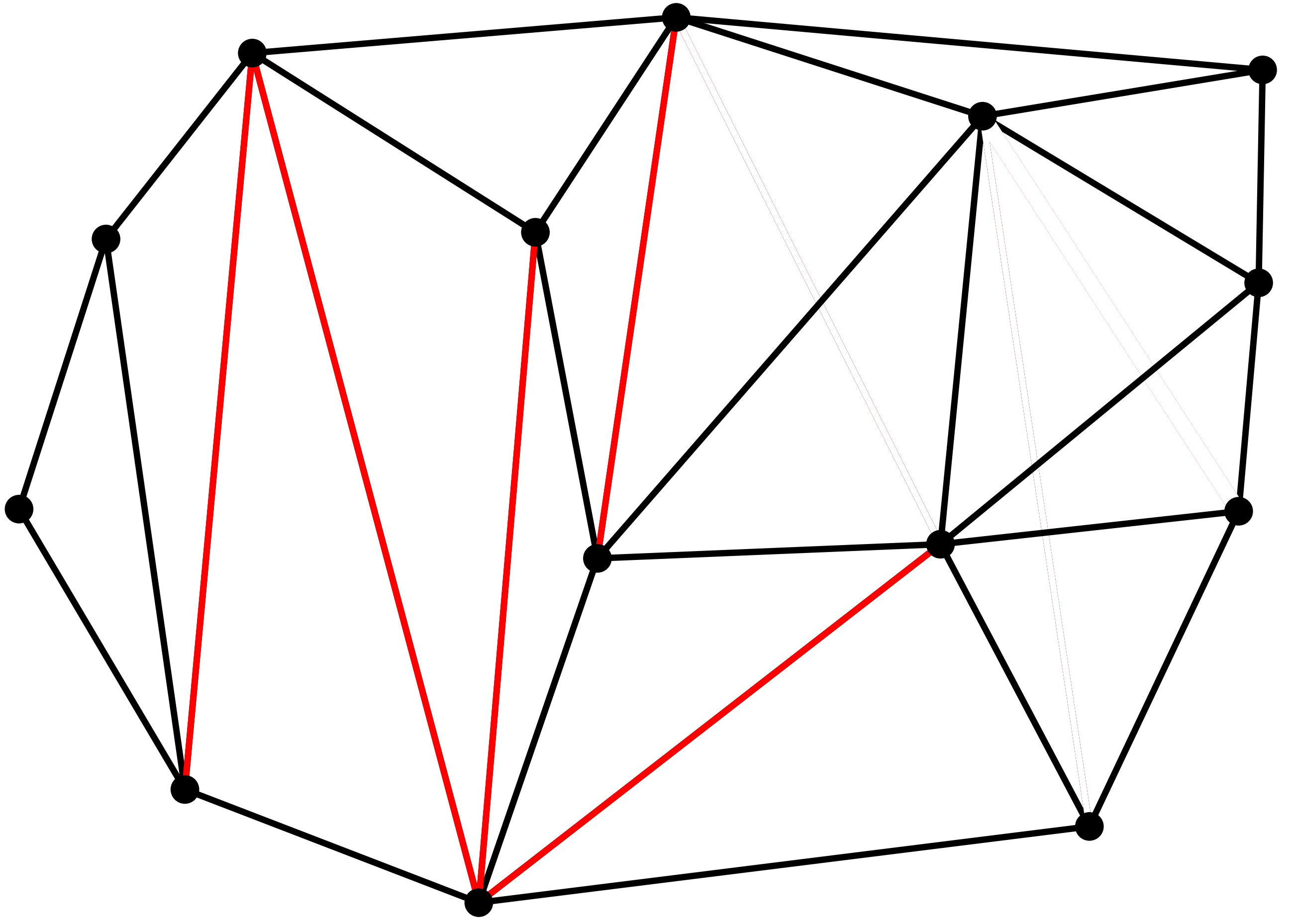
Delaunay Triangulation: Diagonal flipping



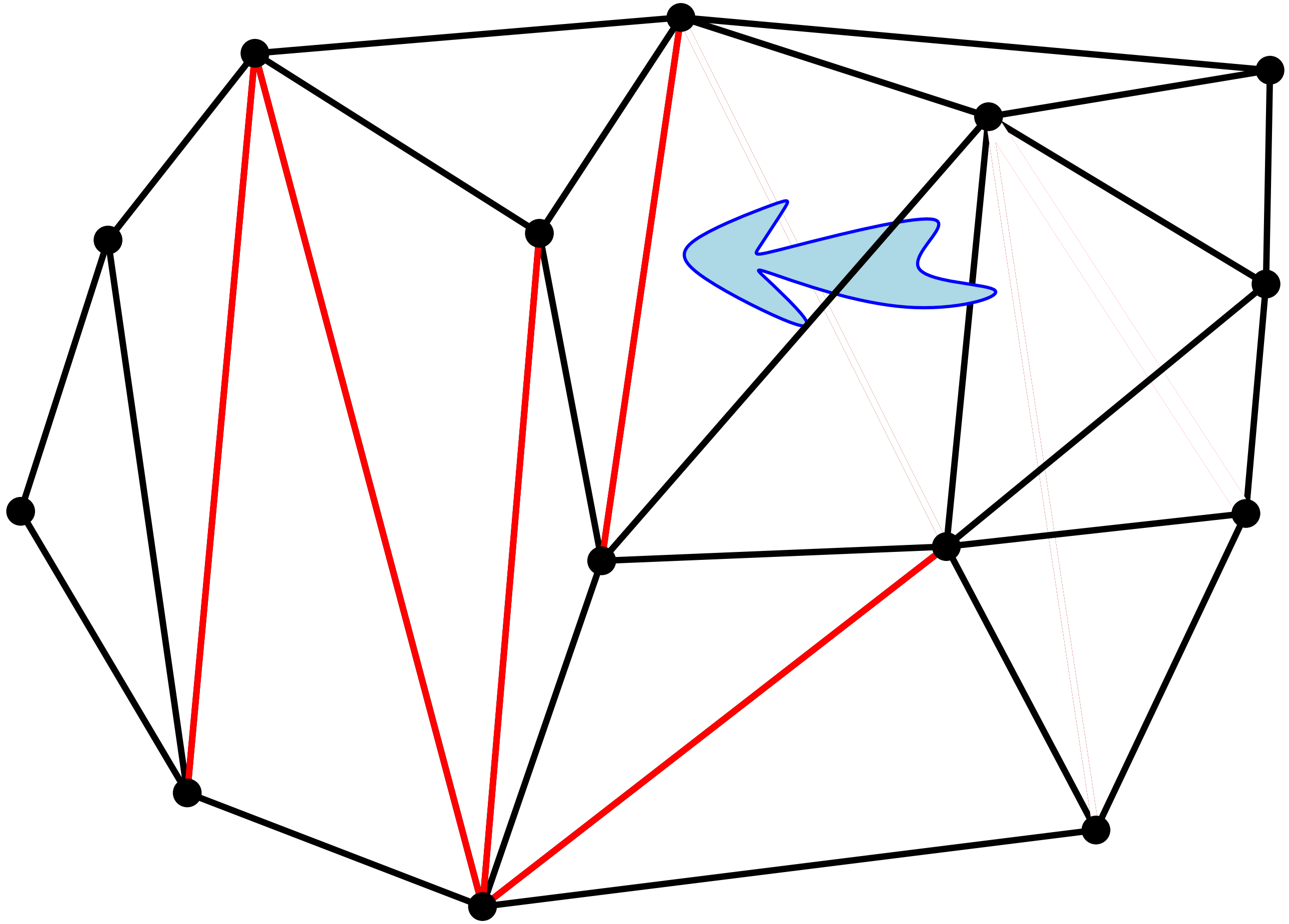
Delaunay Triangulation: Diagonal flipping



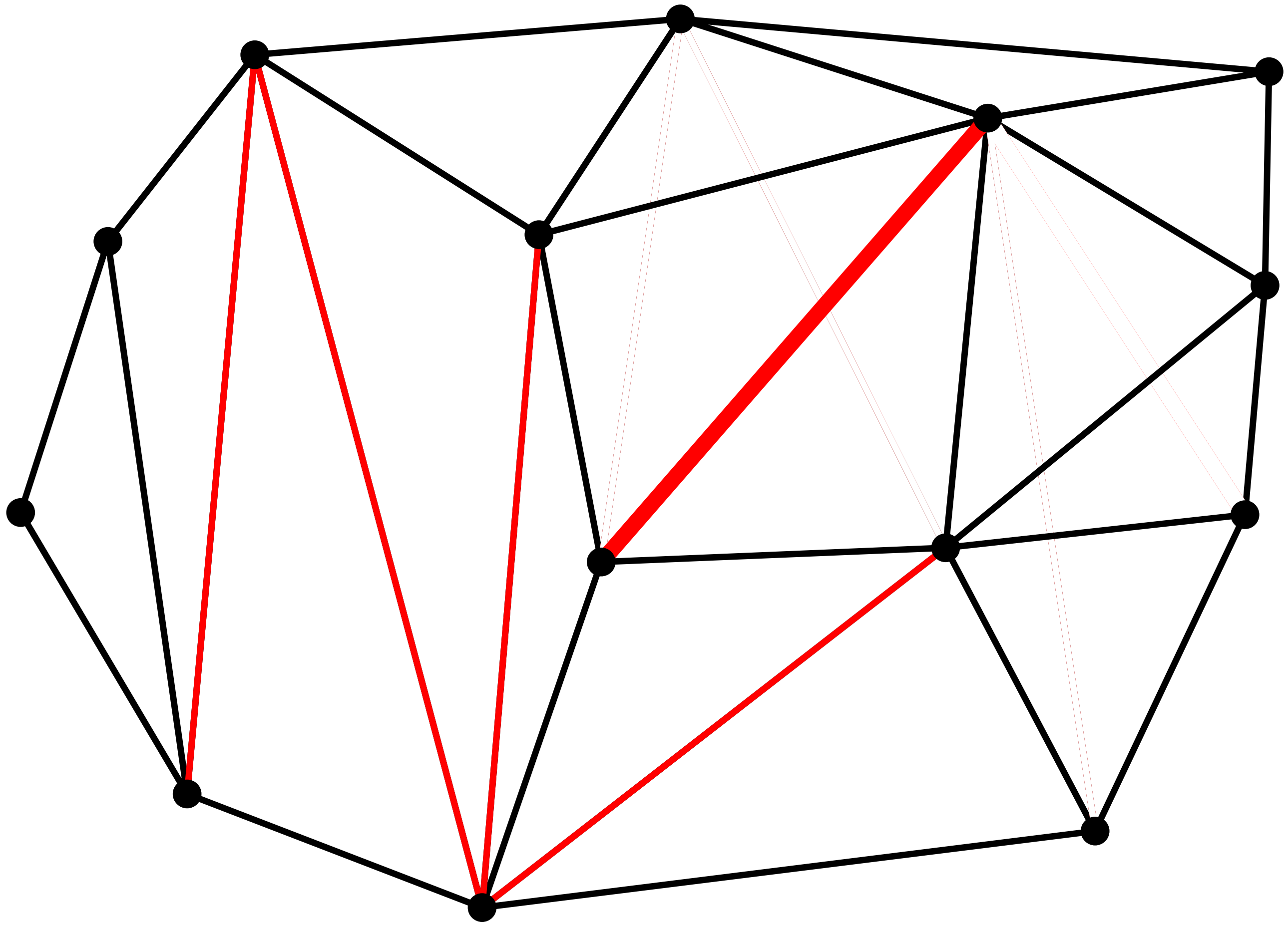
Delaunay Triangulation: Diagonal flipping



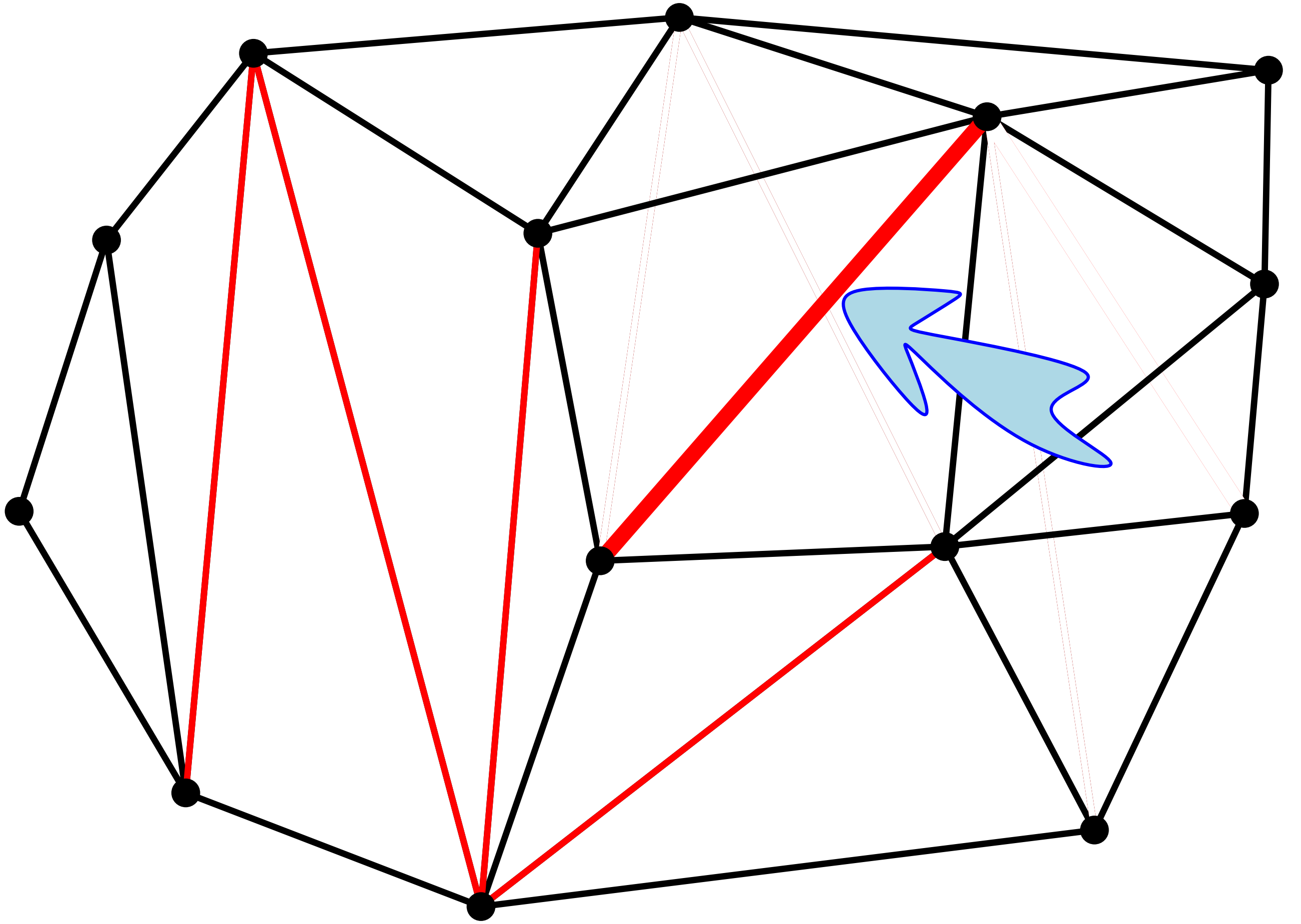
Delaunay Triangulation: Diagonal flipping



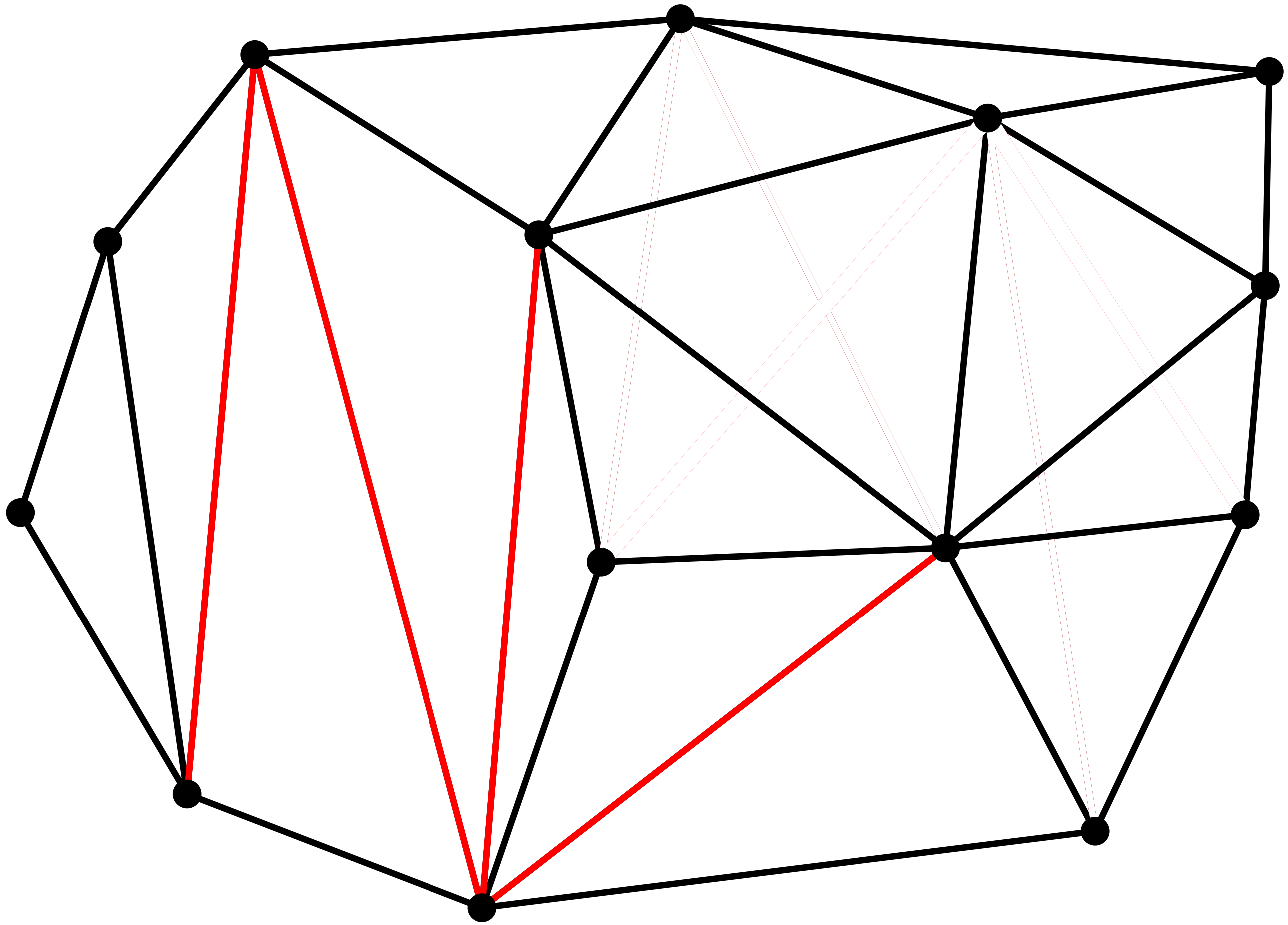
Delaunay Triangulation: Diagonal flipping



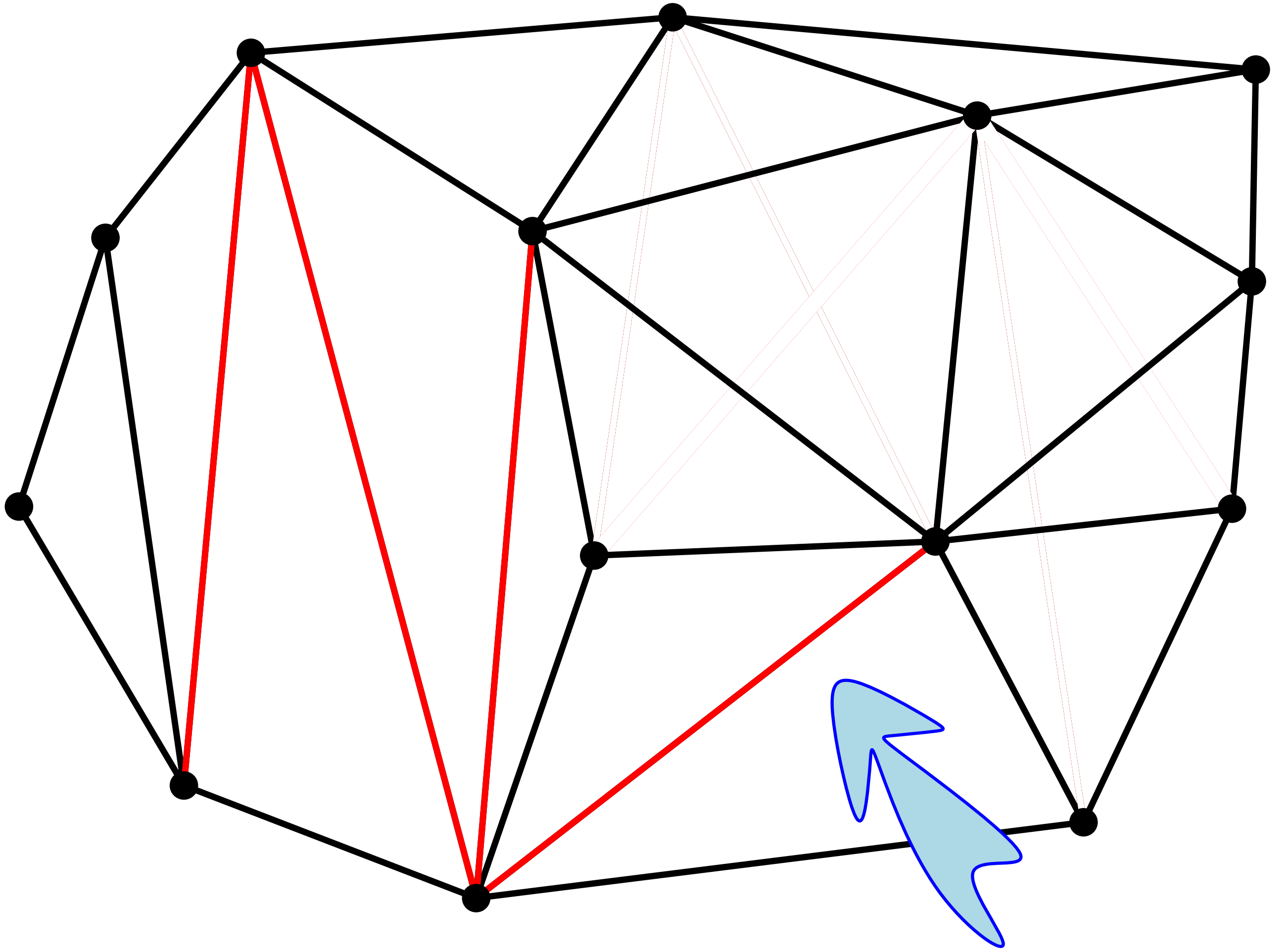
Delaunay Triangulation: Diagonal flipping



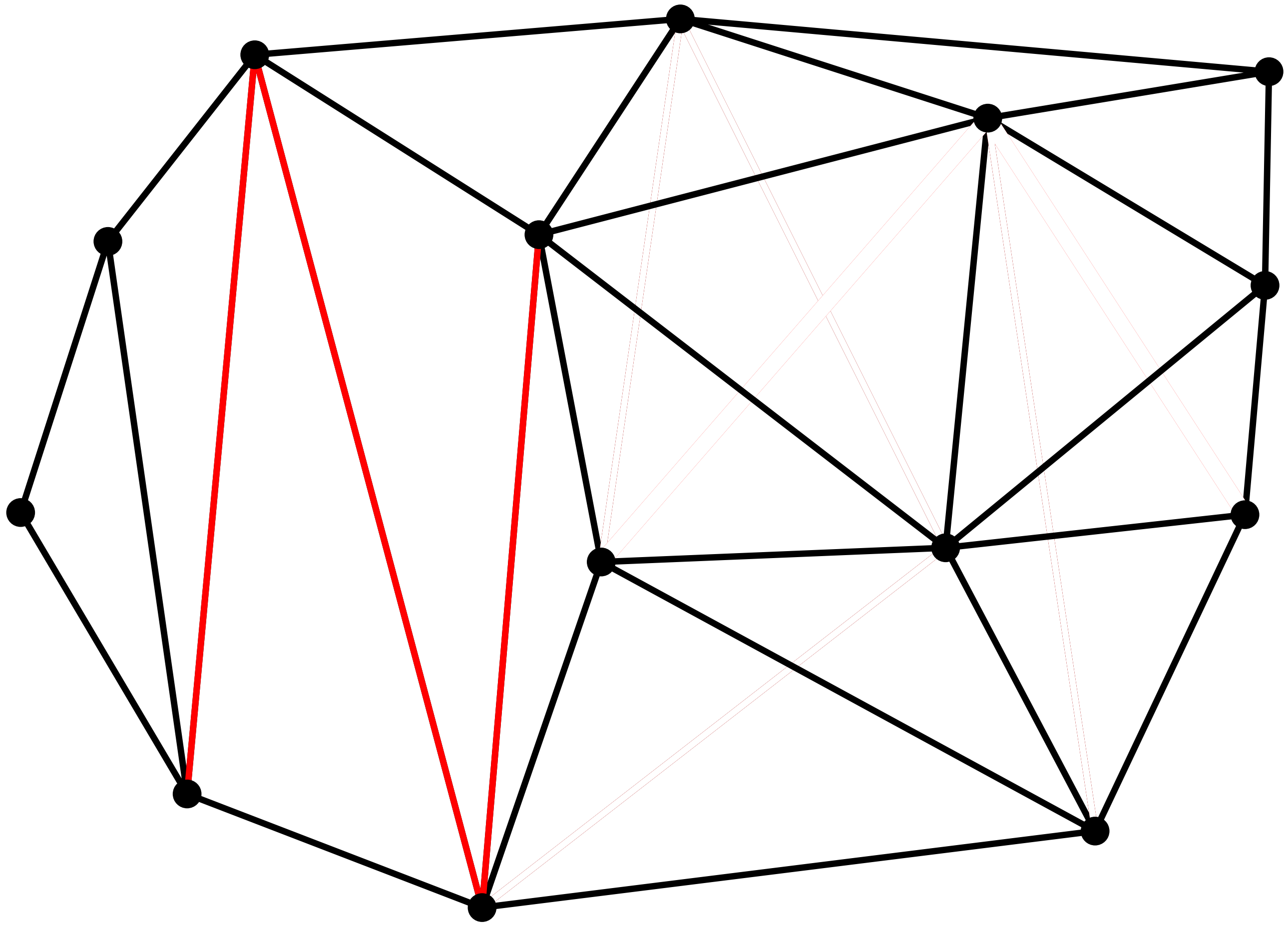
Delaunay Triangulation: Diagonal flipping



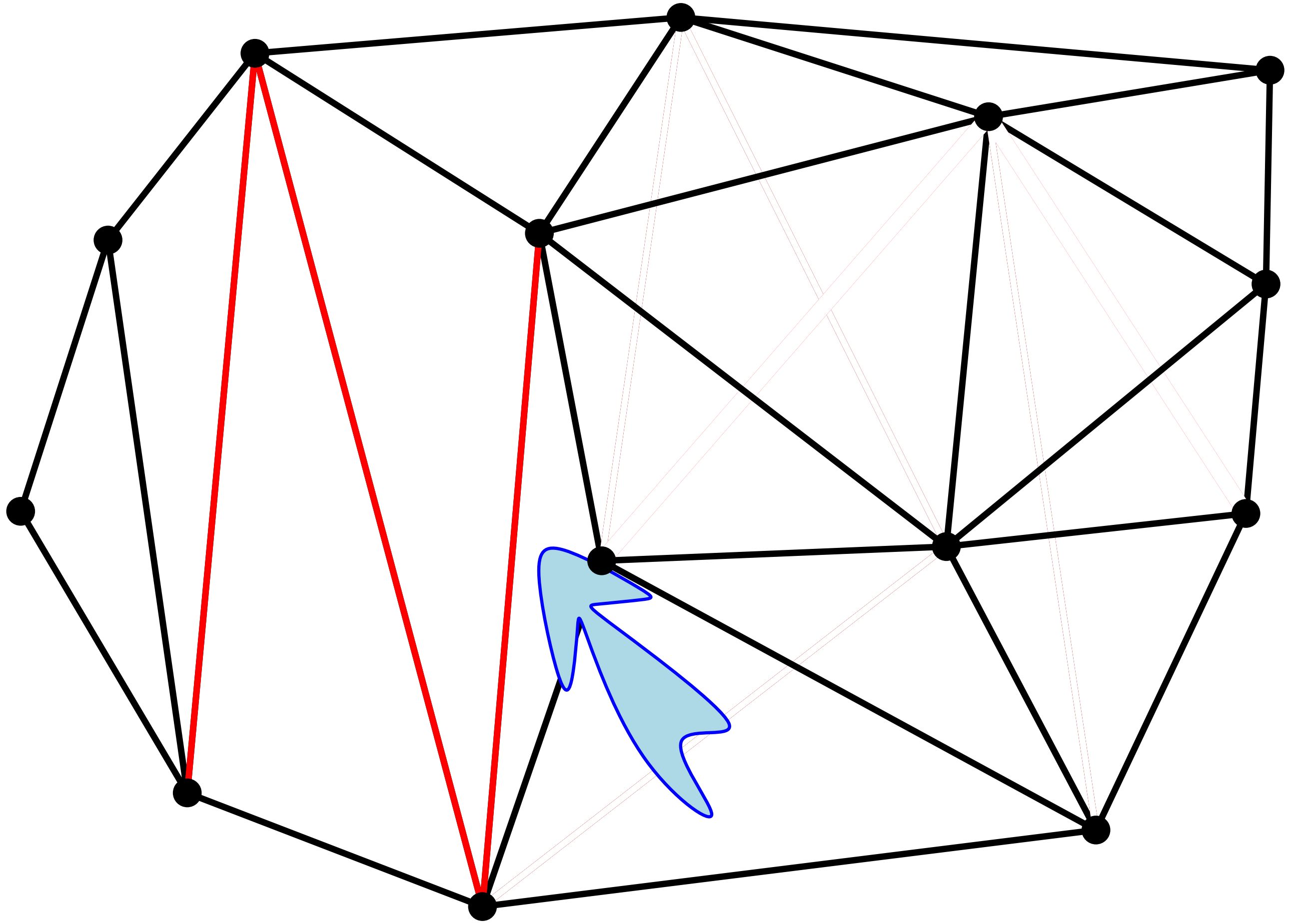
Delaunay Triangulation: Diagonal flipping



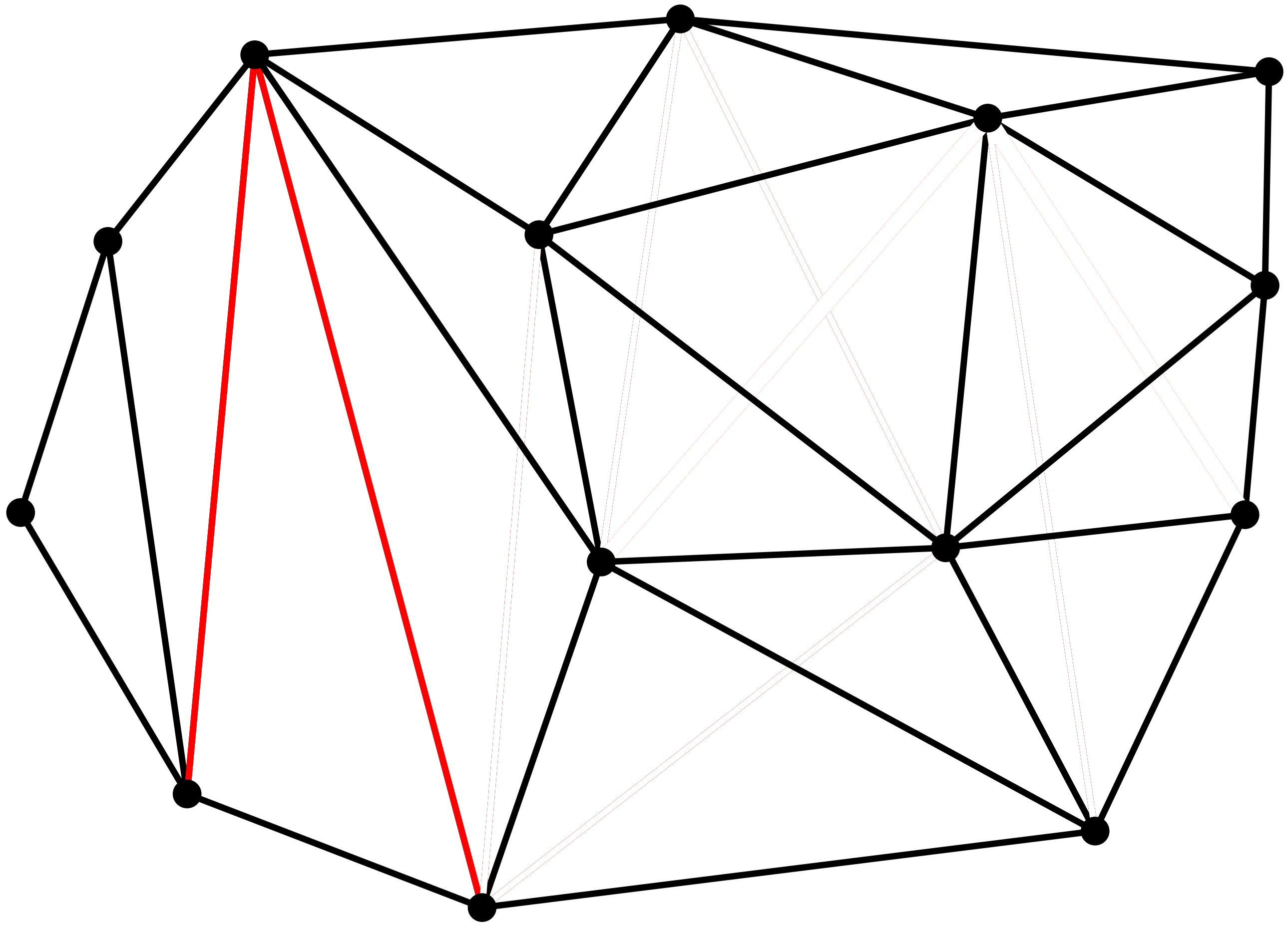
Delaunay Triangulation: Diagonal flipping



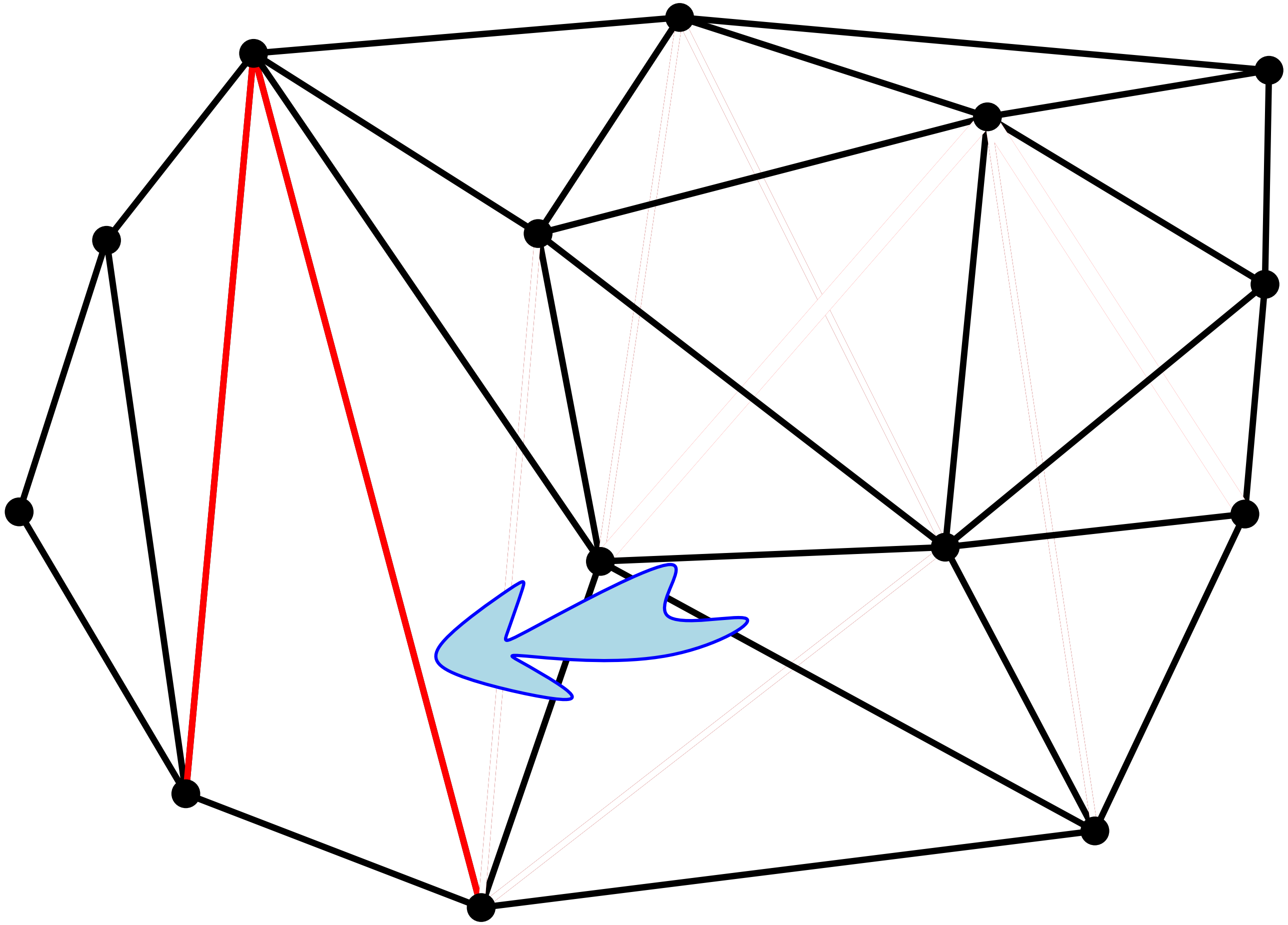
Delaunay Triangulation: Diagonal flipping



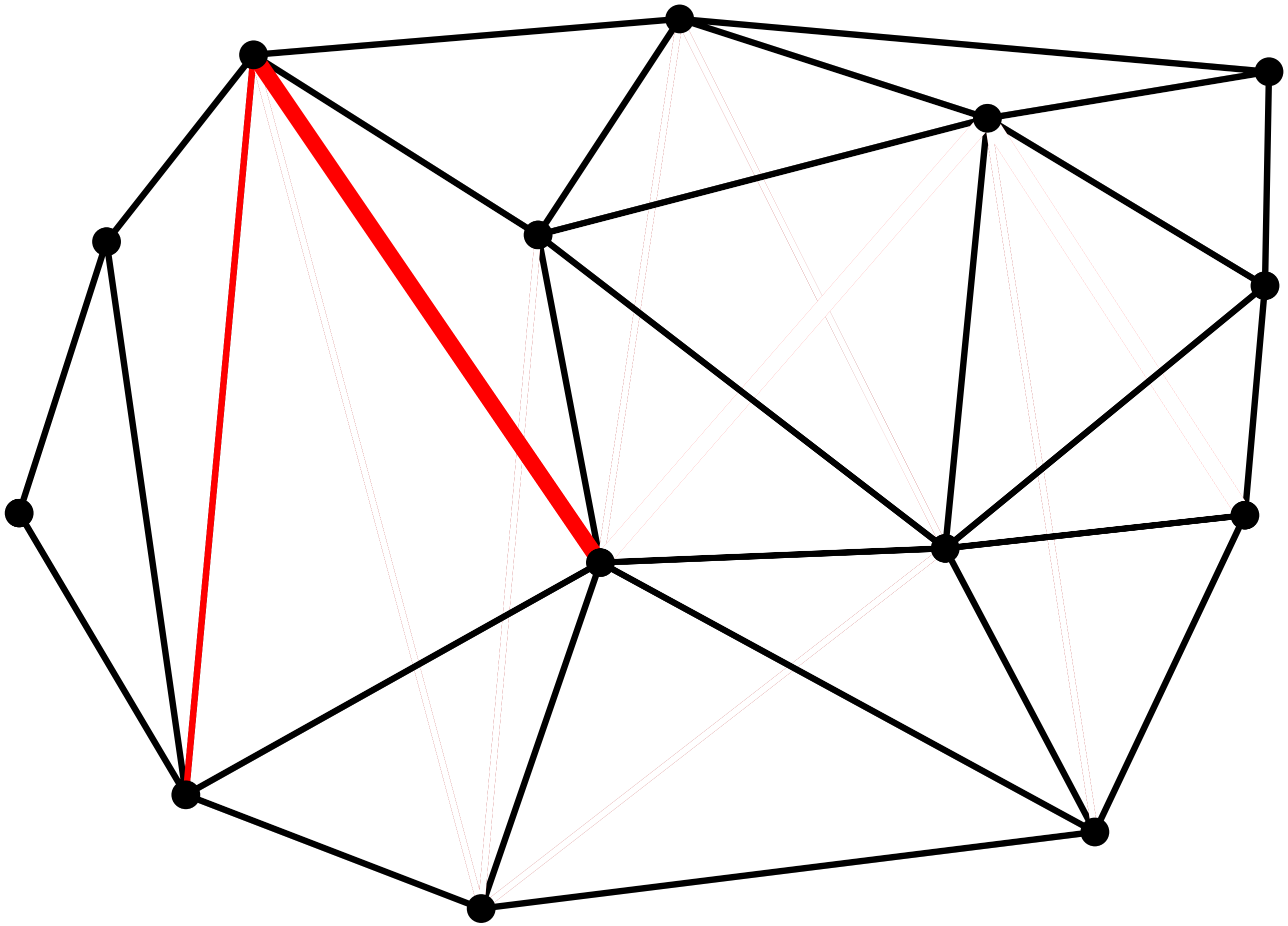
Delaunay Triangulation: Diagonal flipping



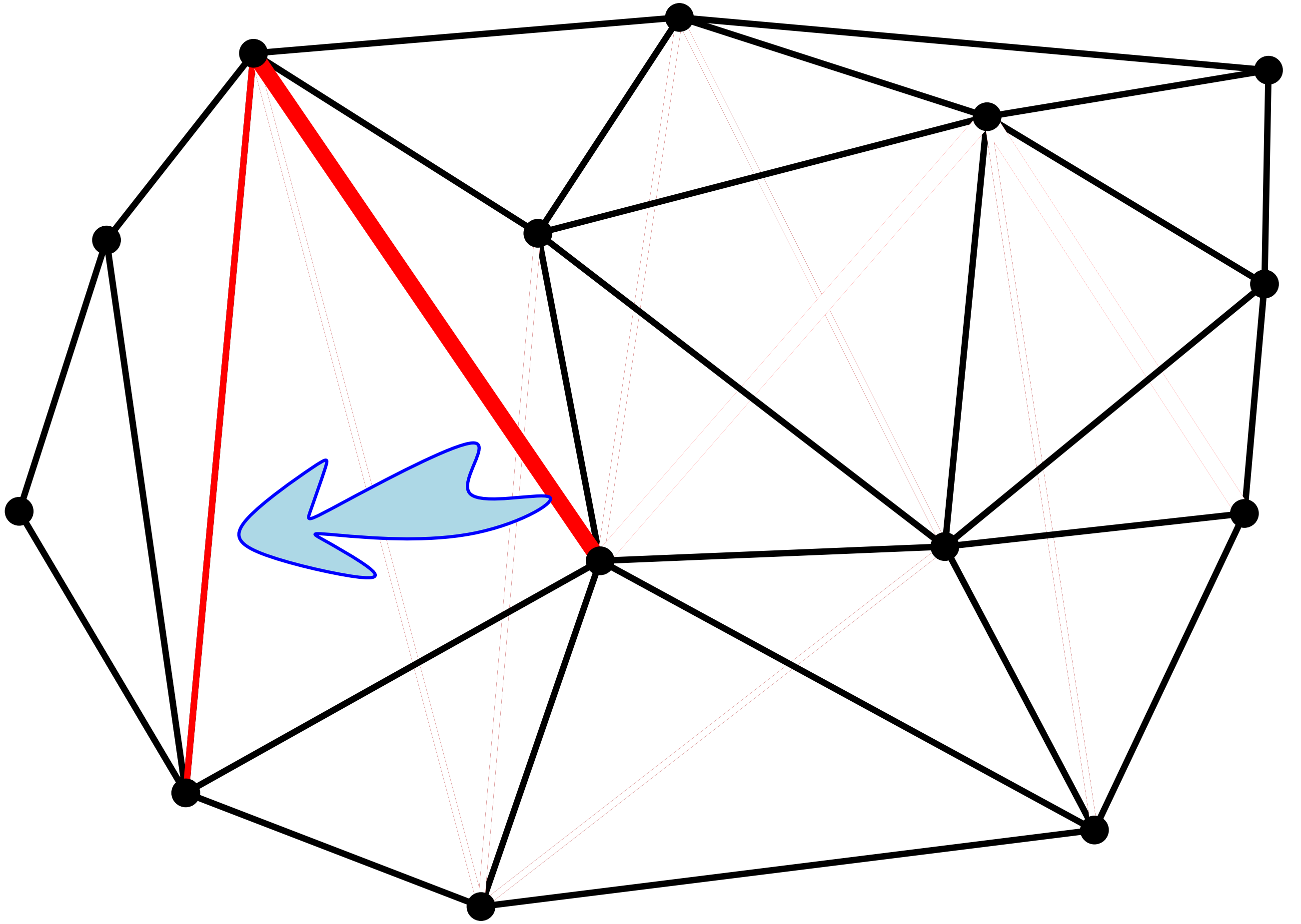
Delaunay Triangulation: Diagonal flipping



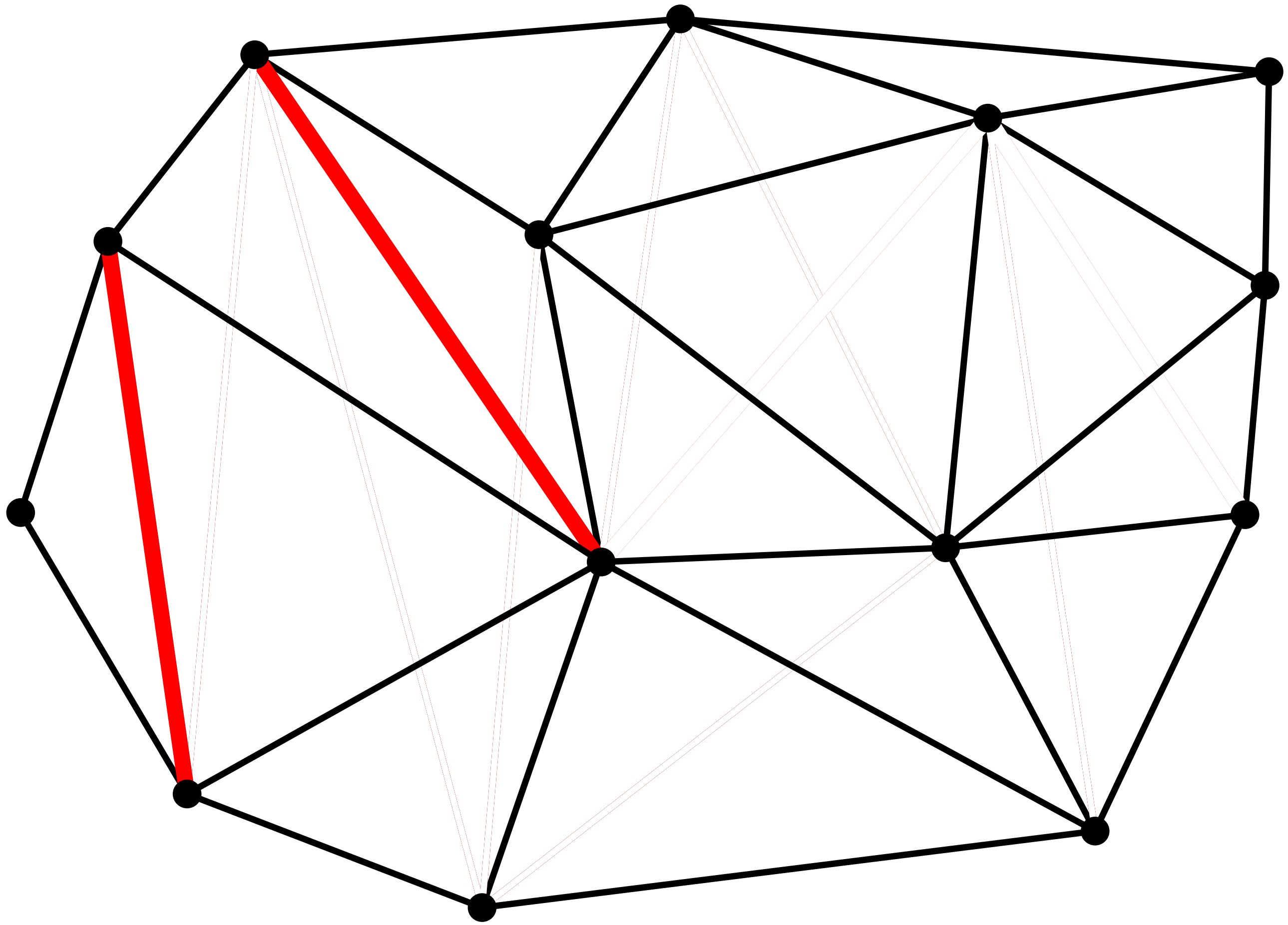
Delaunay Triangulation: Diagonal flipping



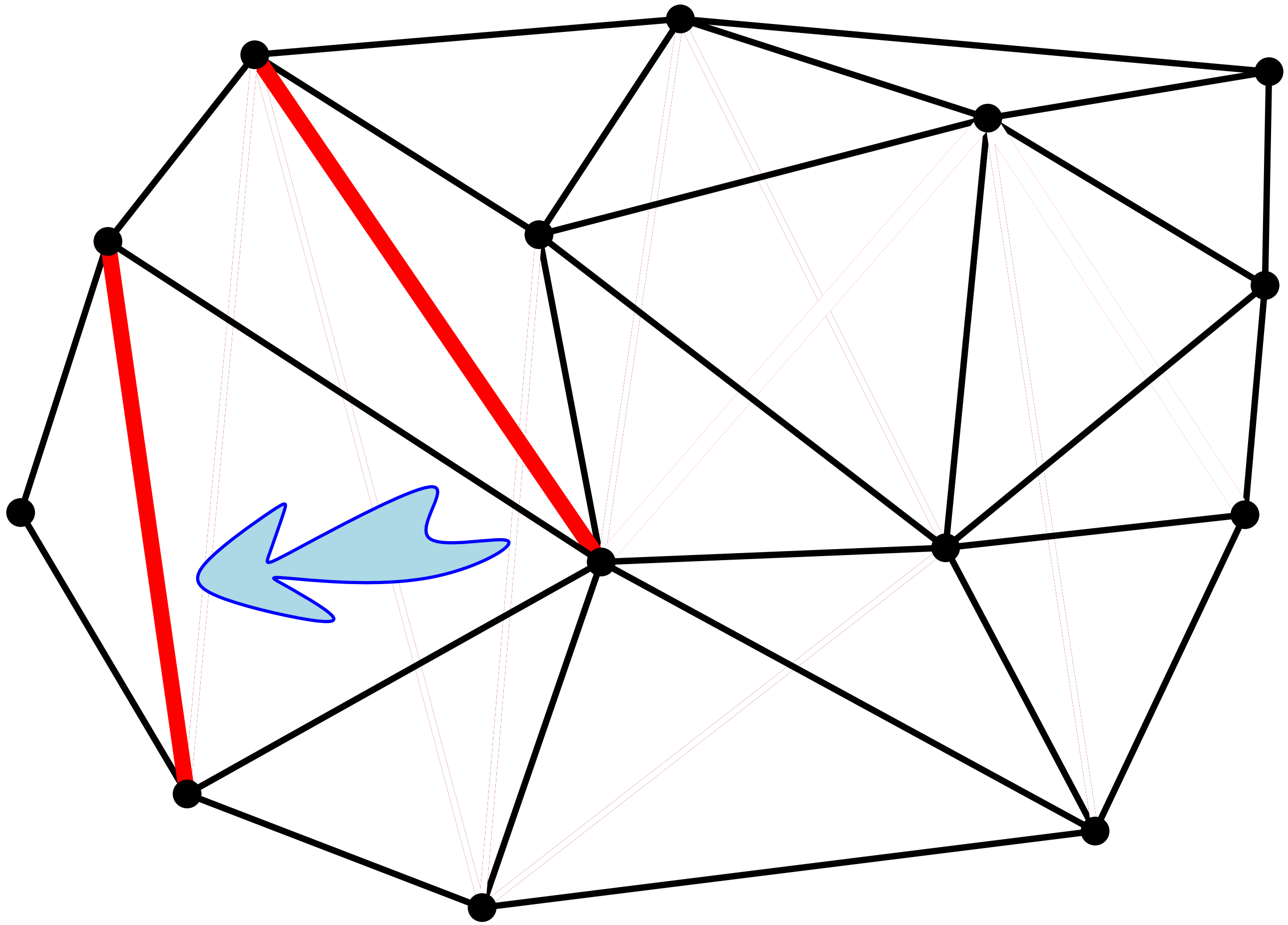
Delaunay Triangulation: Diagonal flipping



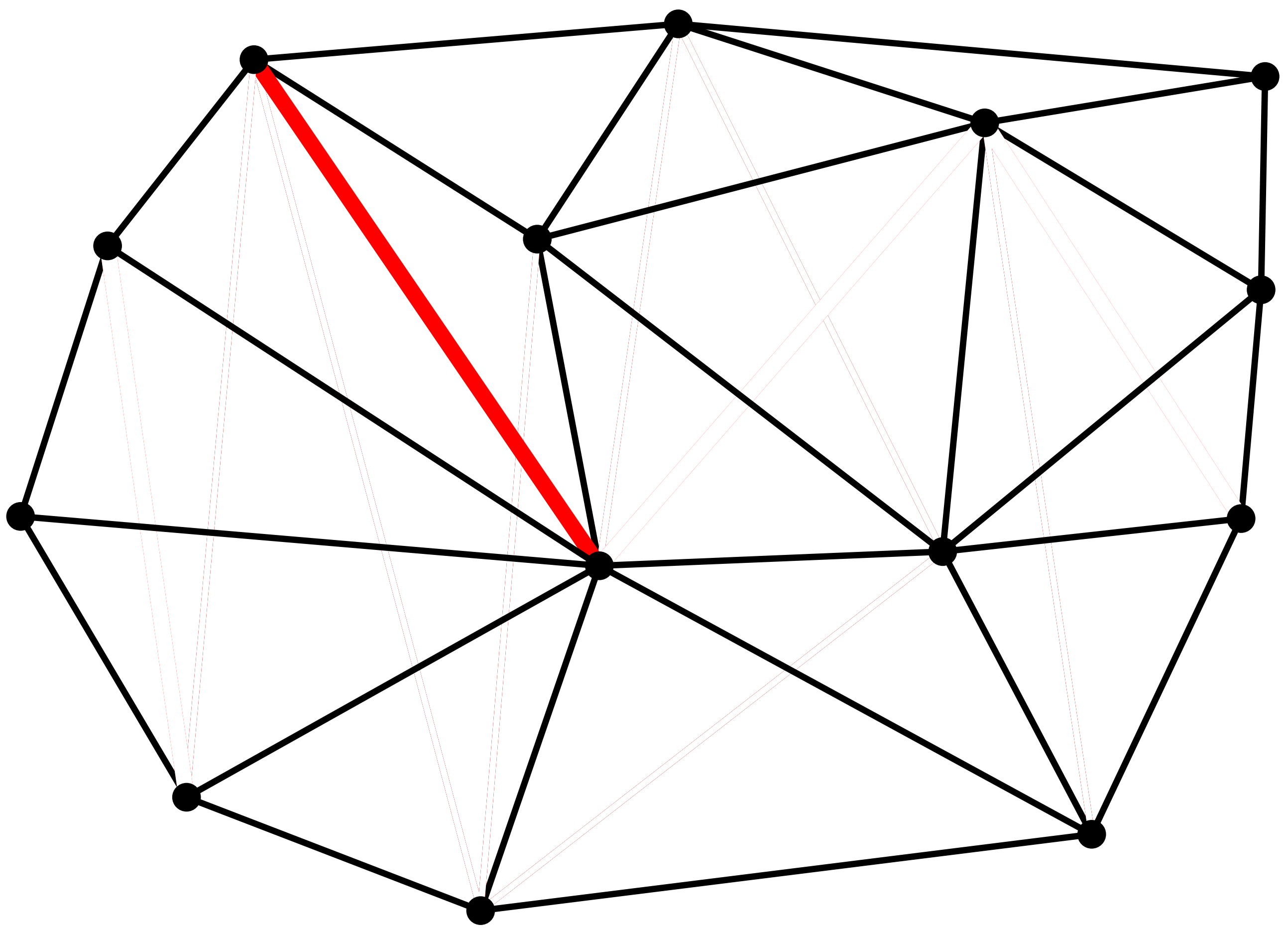
Delaunay Triangulation: Diagonal flipping



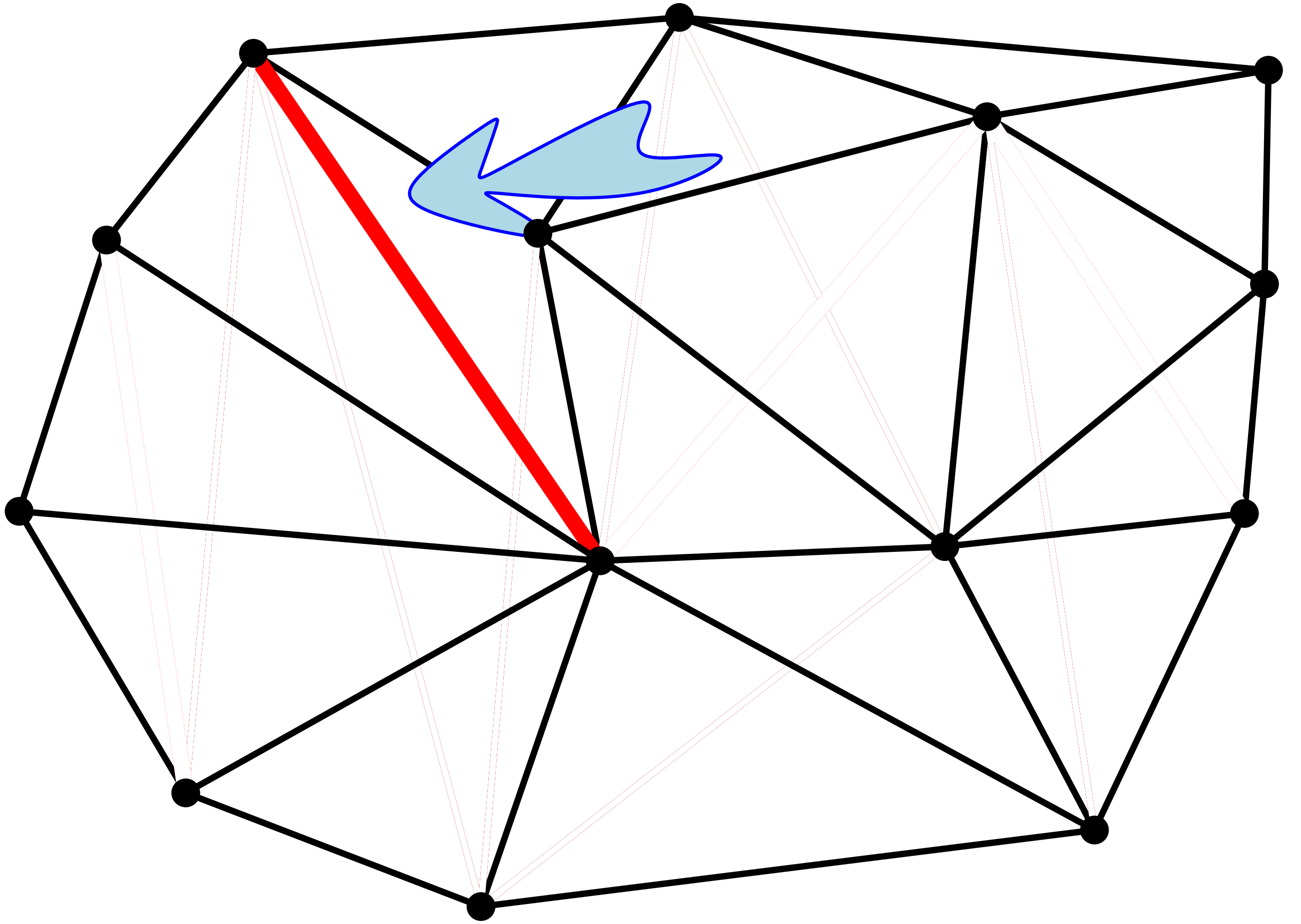
Delaunay Triangulation: Diagonal flipping



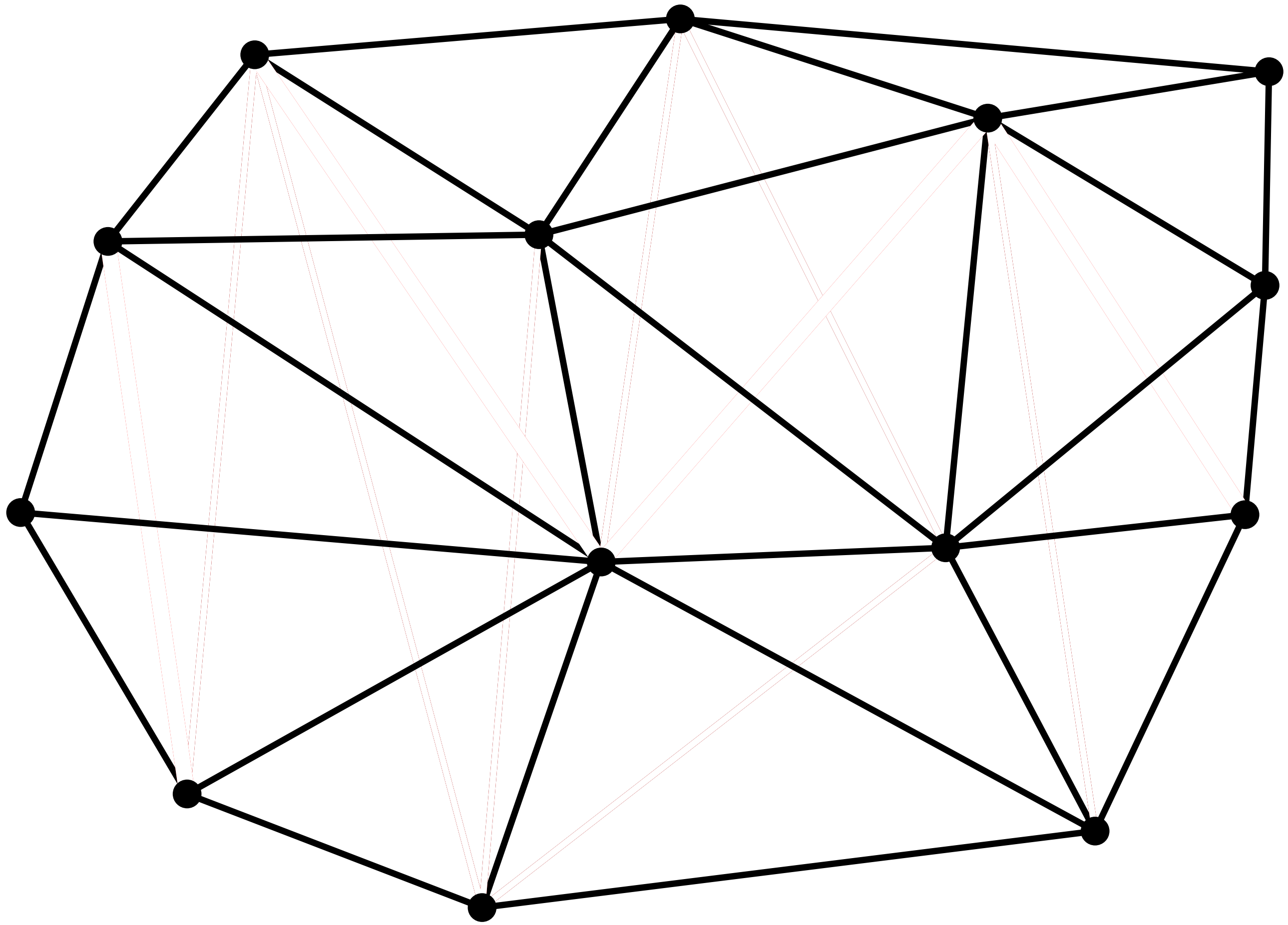
Delaunay Triangulation: Diagonal flipping



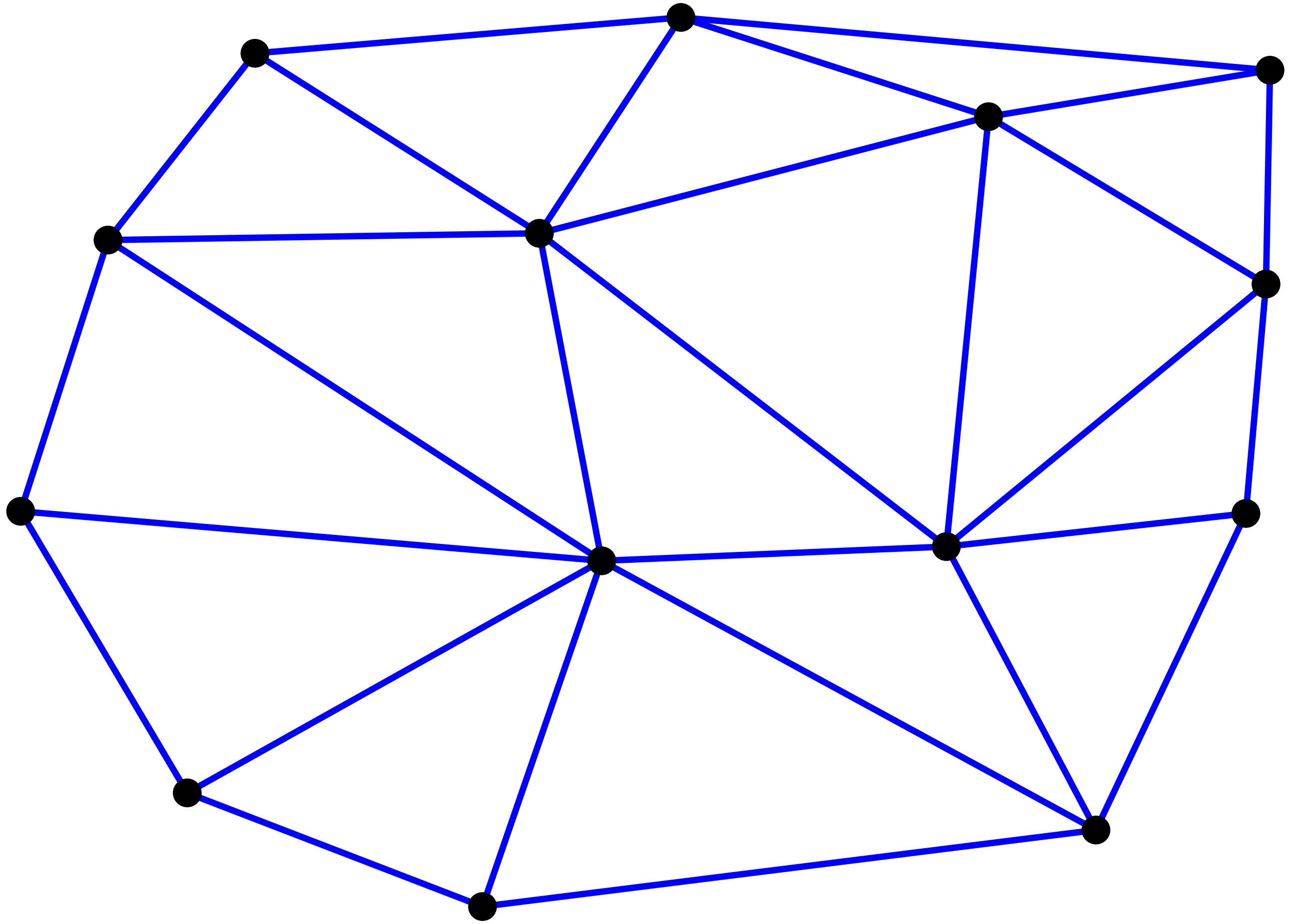
Delaunay Triangulation: Diagonal flipping



Delaunay Triangulation: Diagonal flipping



Delaunay Triangulation: Diagonal flipping



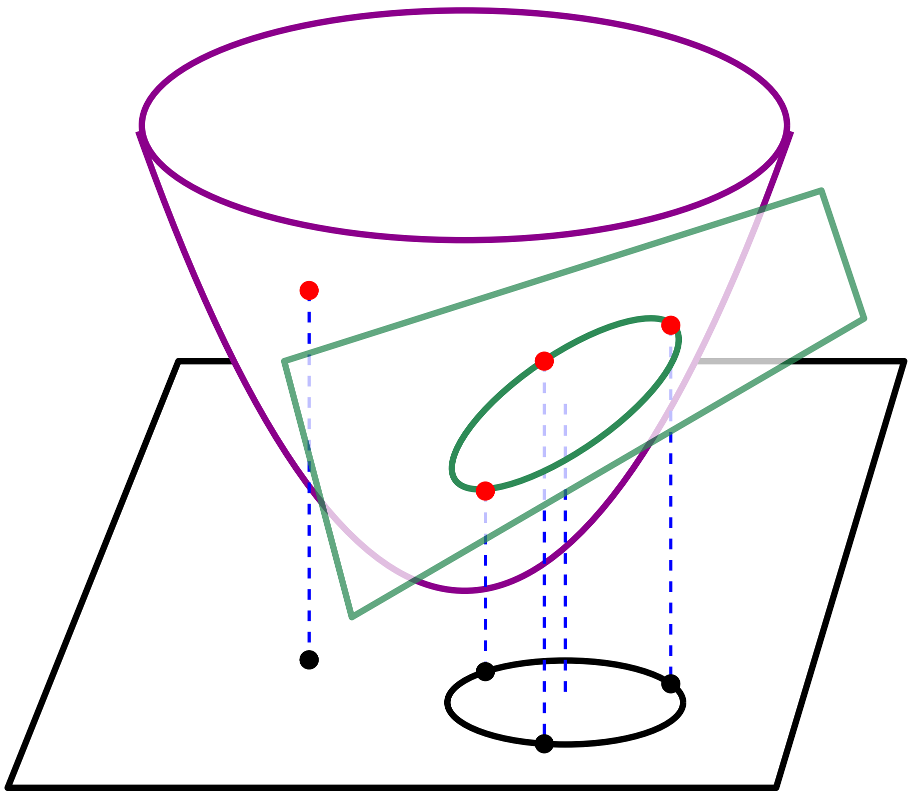
Delaunay is obtained

Delaunay Triangulation: Diagonal flipping

Complexity ?

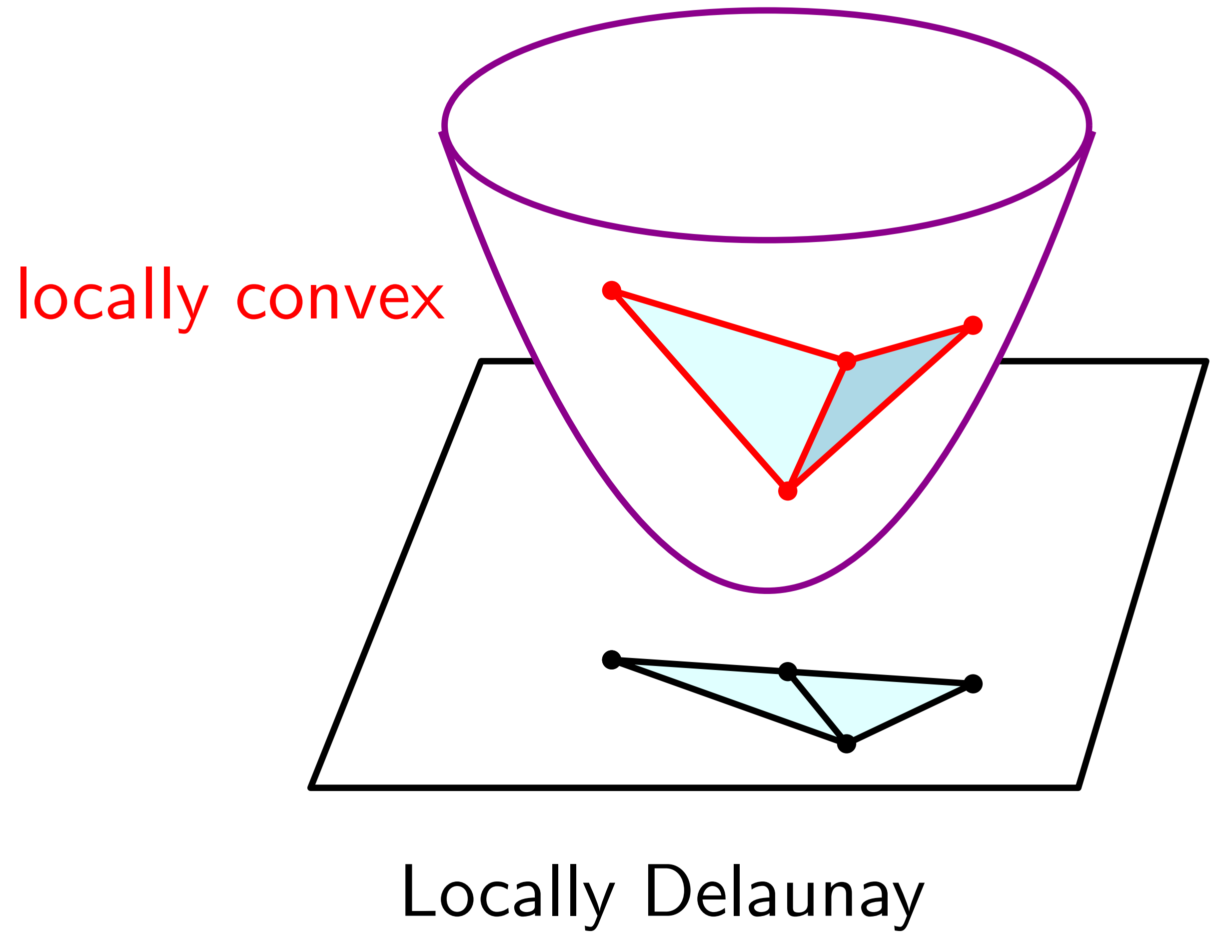
Delaunay Triangulation: Diagonal flipping

Complexity ?



Delaunay Triangulation: Diagonal flipping

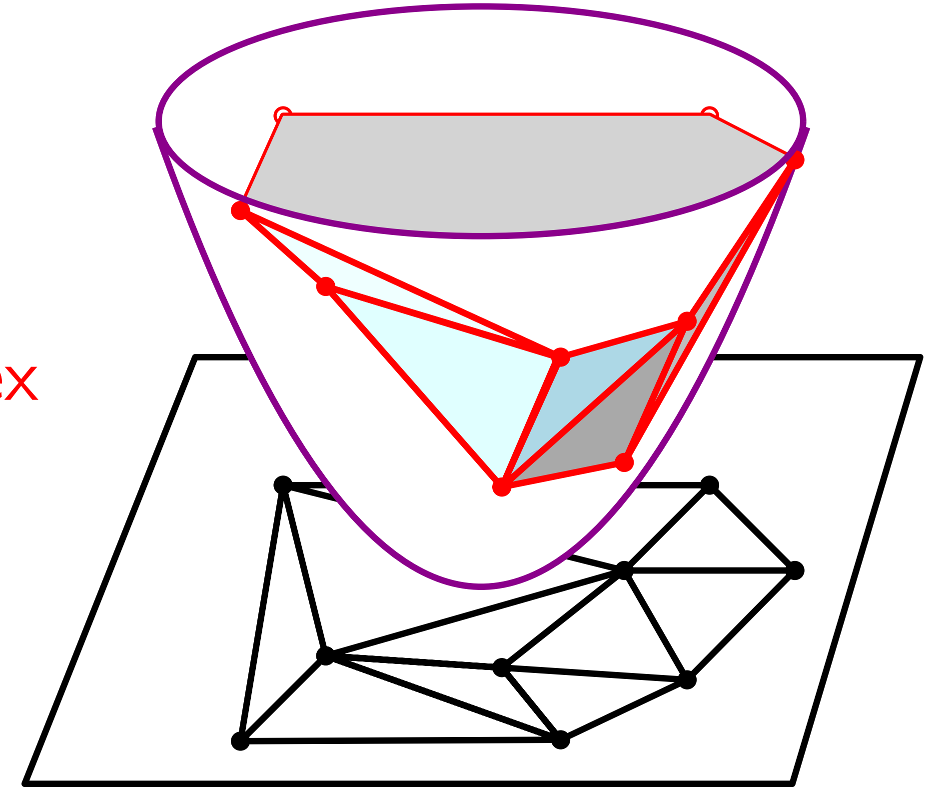
Complexity ?



Delaunay Triangulation: Diagonal flipping

Complexity ?

Convex

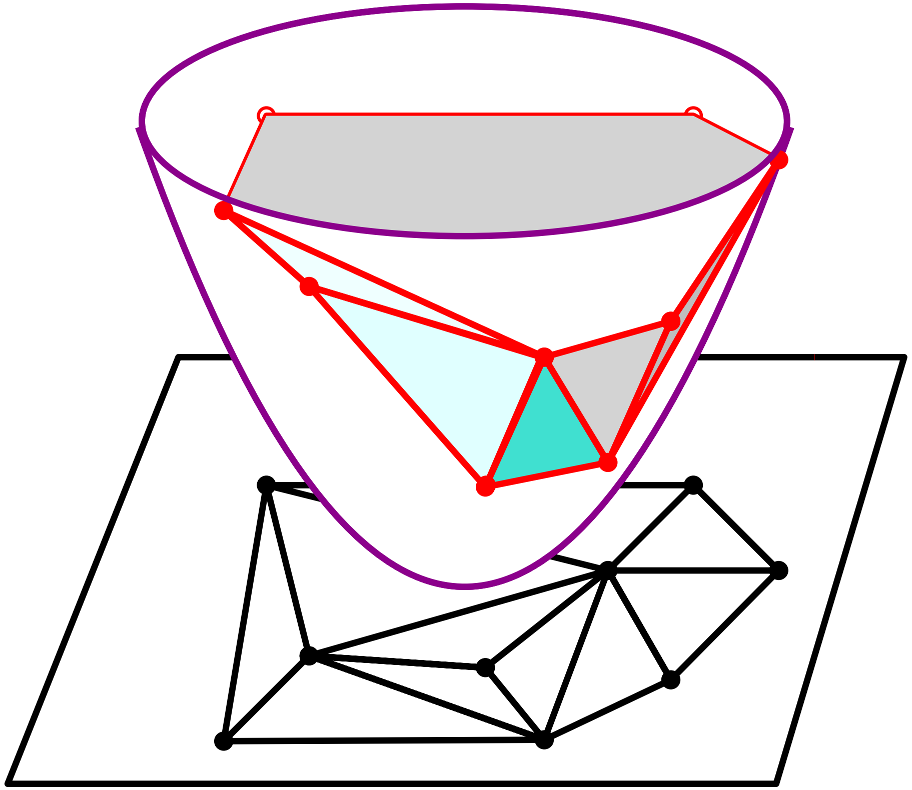


Delaunay

Delaunay Triangulation: Diagonal flipping

Complexity ?

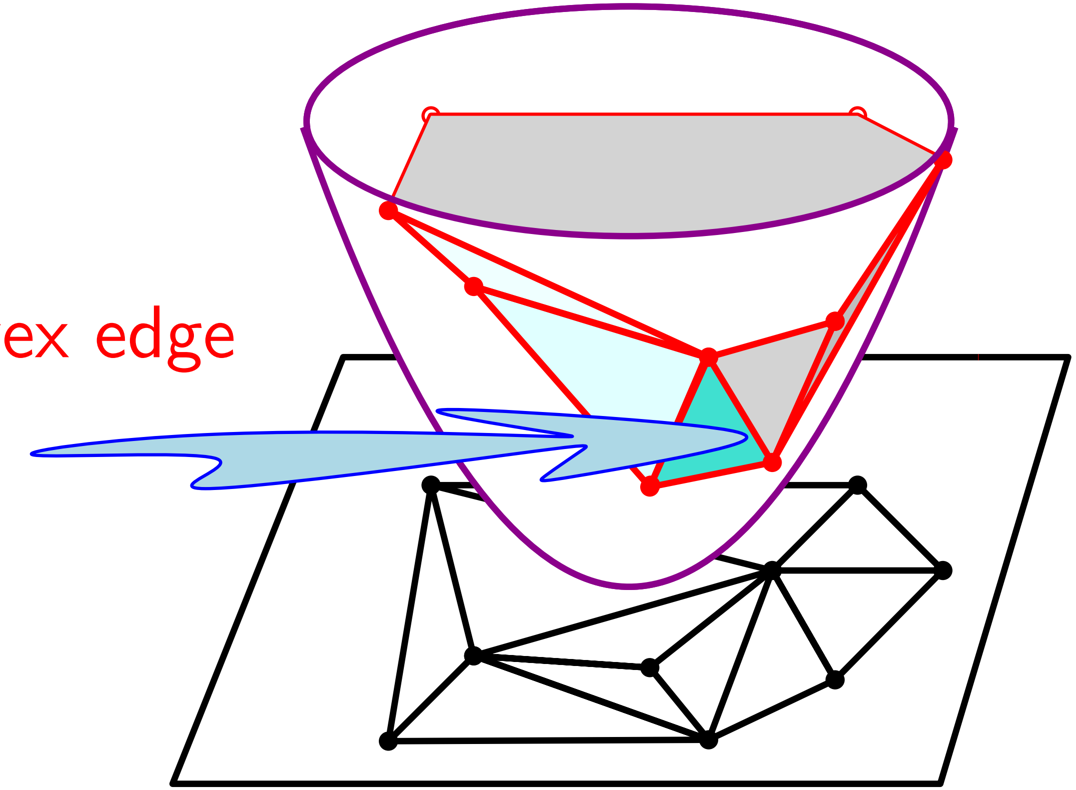
Non convex



Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex edge



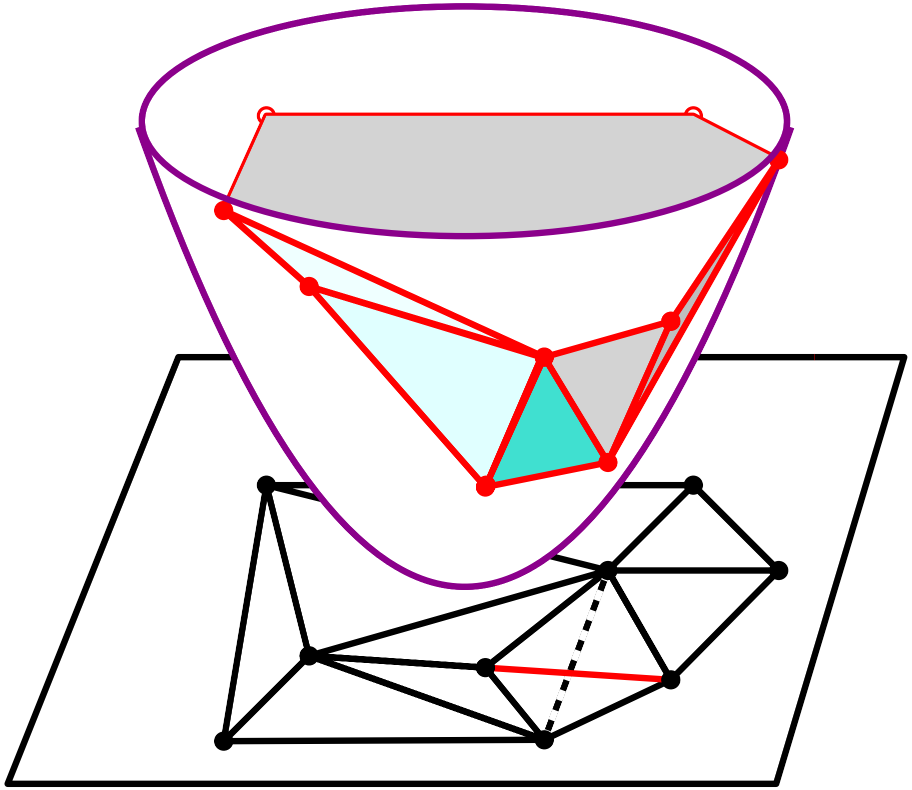
Non Delaunay

Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip

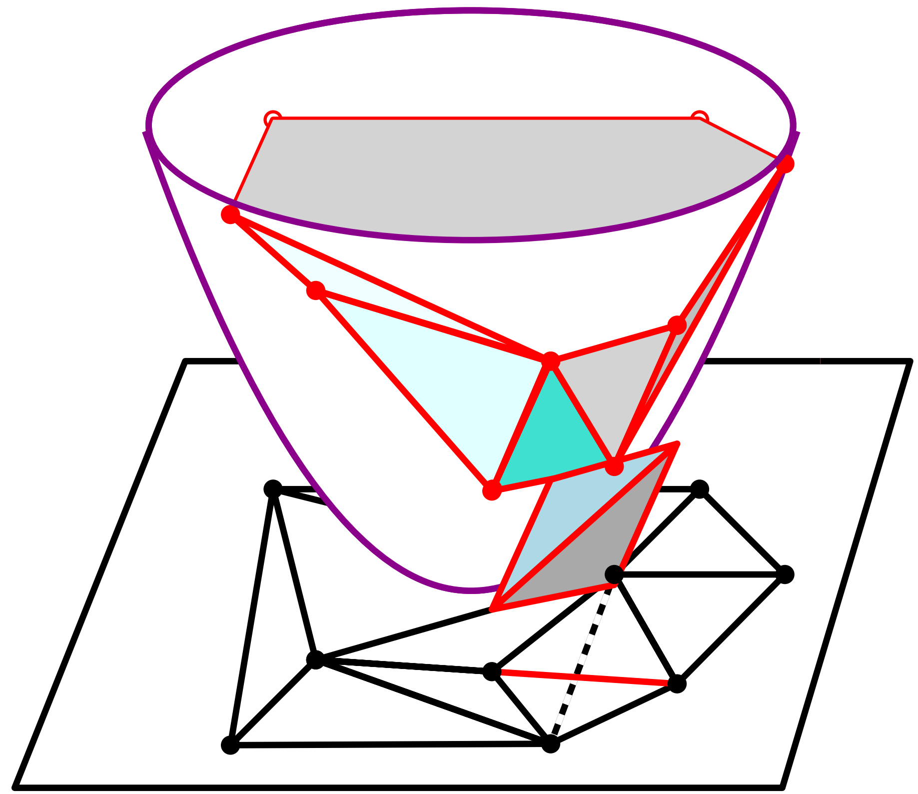


Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip

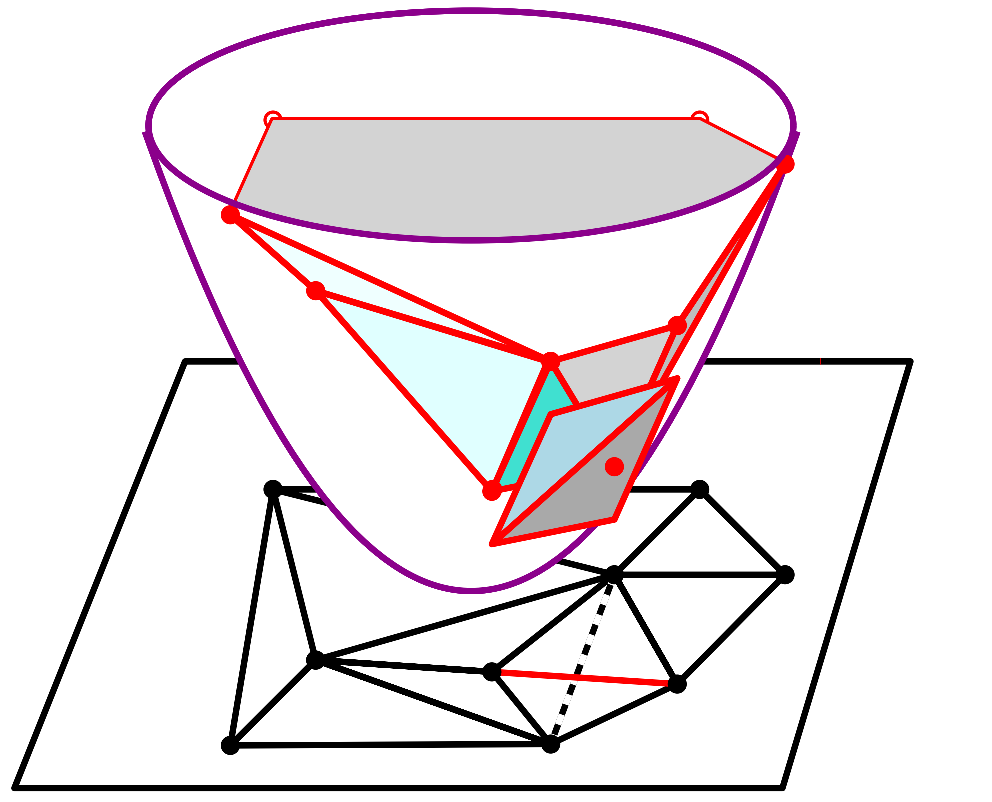


Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip

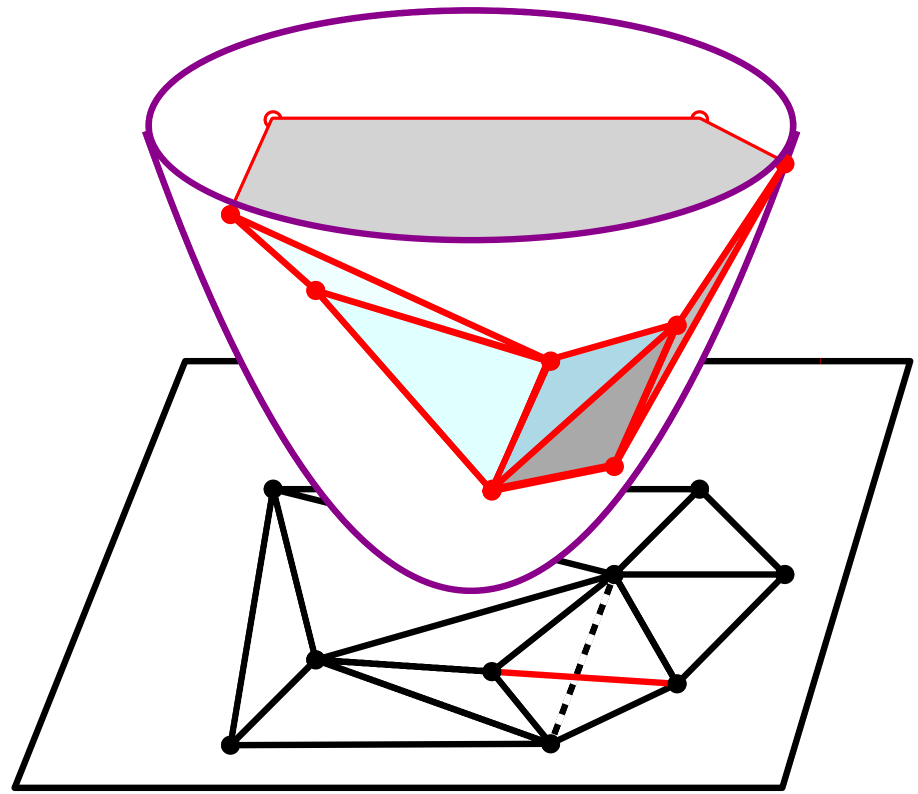


Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip



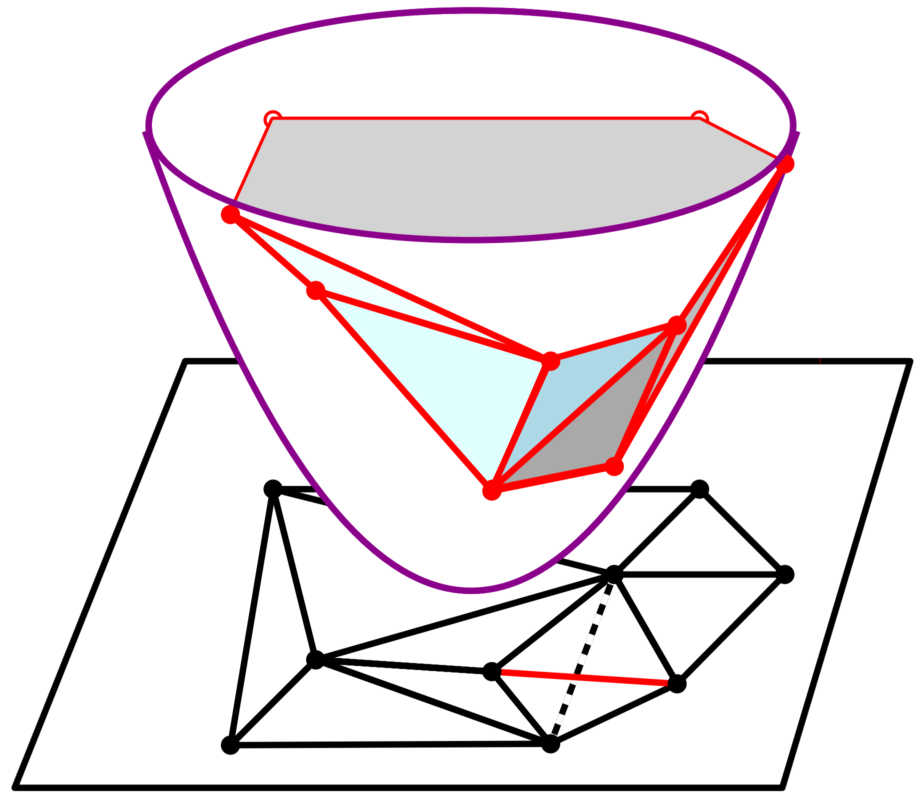
Non Delaunay

Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip



An hidden edge cannot be visible again

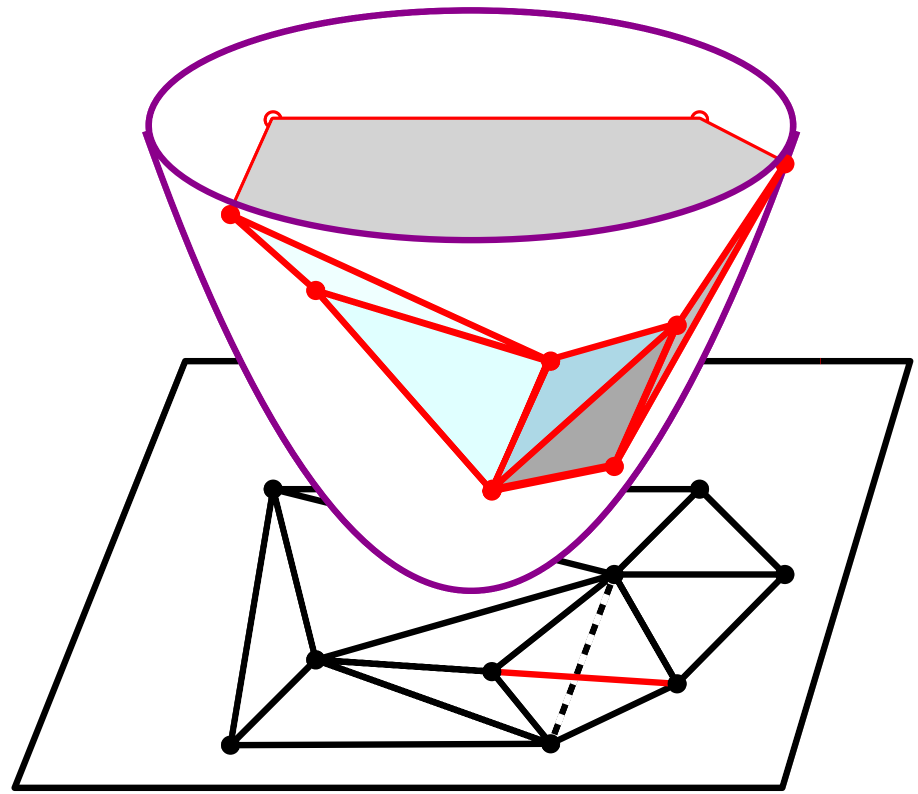
Non Delaunay

Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip



An hidden edge cannot be visible again

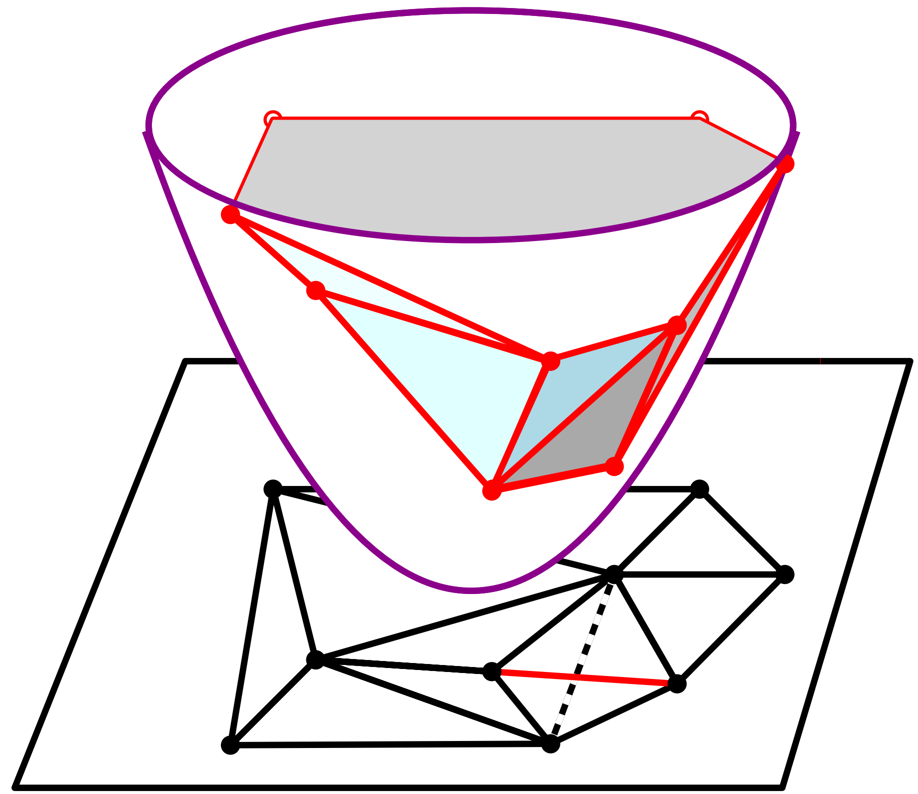
At most $\frac{n(n-1)}{2}$ edges

Delaunay Triangulation: Diagonal flipping

Complexity ?

Non convex

Flip



An hidden edge cannot be visible again

At most $\frac{n(n-1)}{2}$ edges

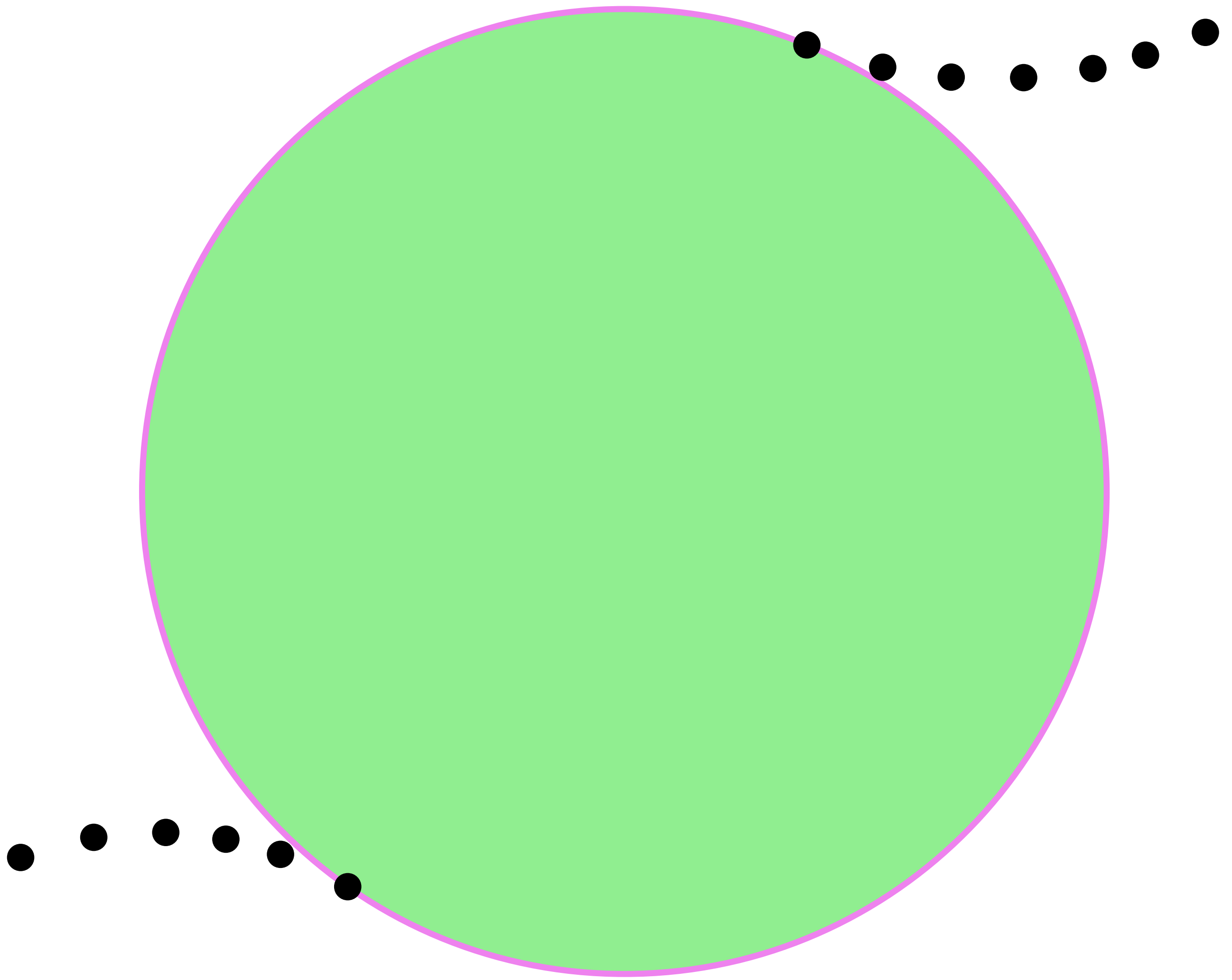
Complexity of diagonal flipping is $O(n^2)$

Delaunay Triangulation: Diagonal flipping

Complexity ?

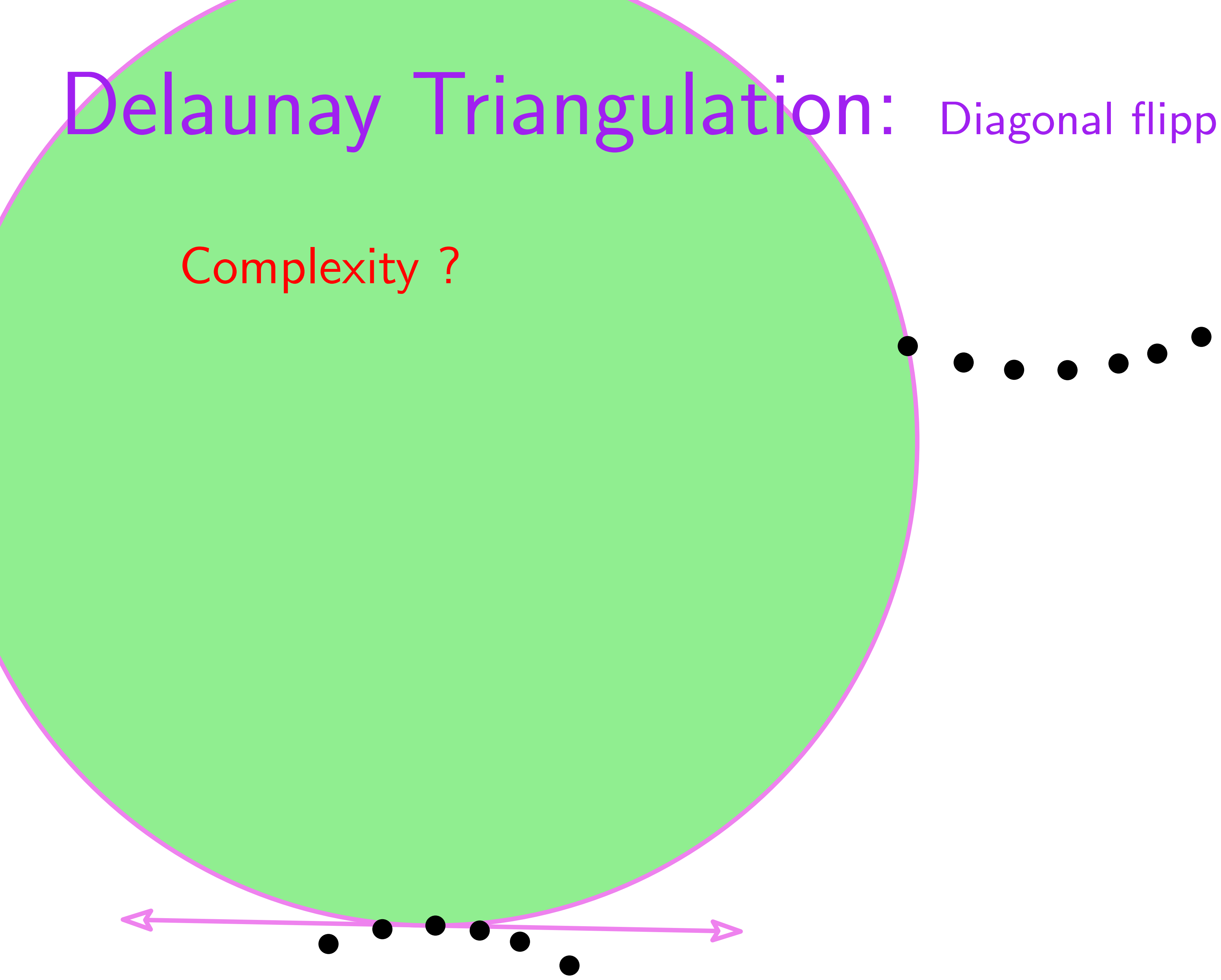
Delaunay Triangulation: Diagonal flipping

Complexity ?



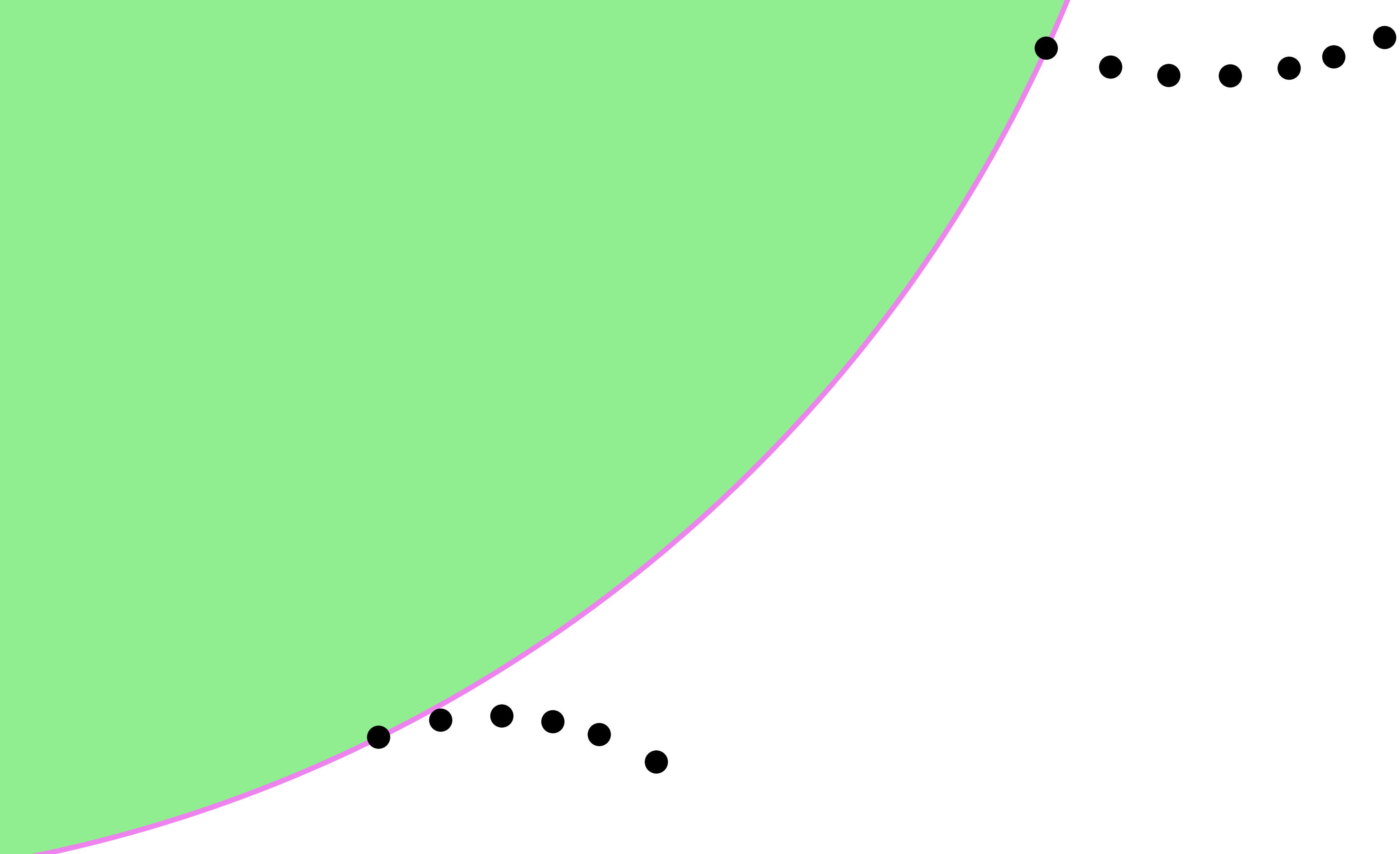
Delaunay Triangulation: Diagonal flipping

Complexity ?



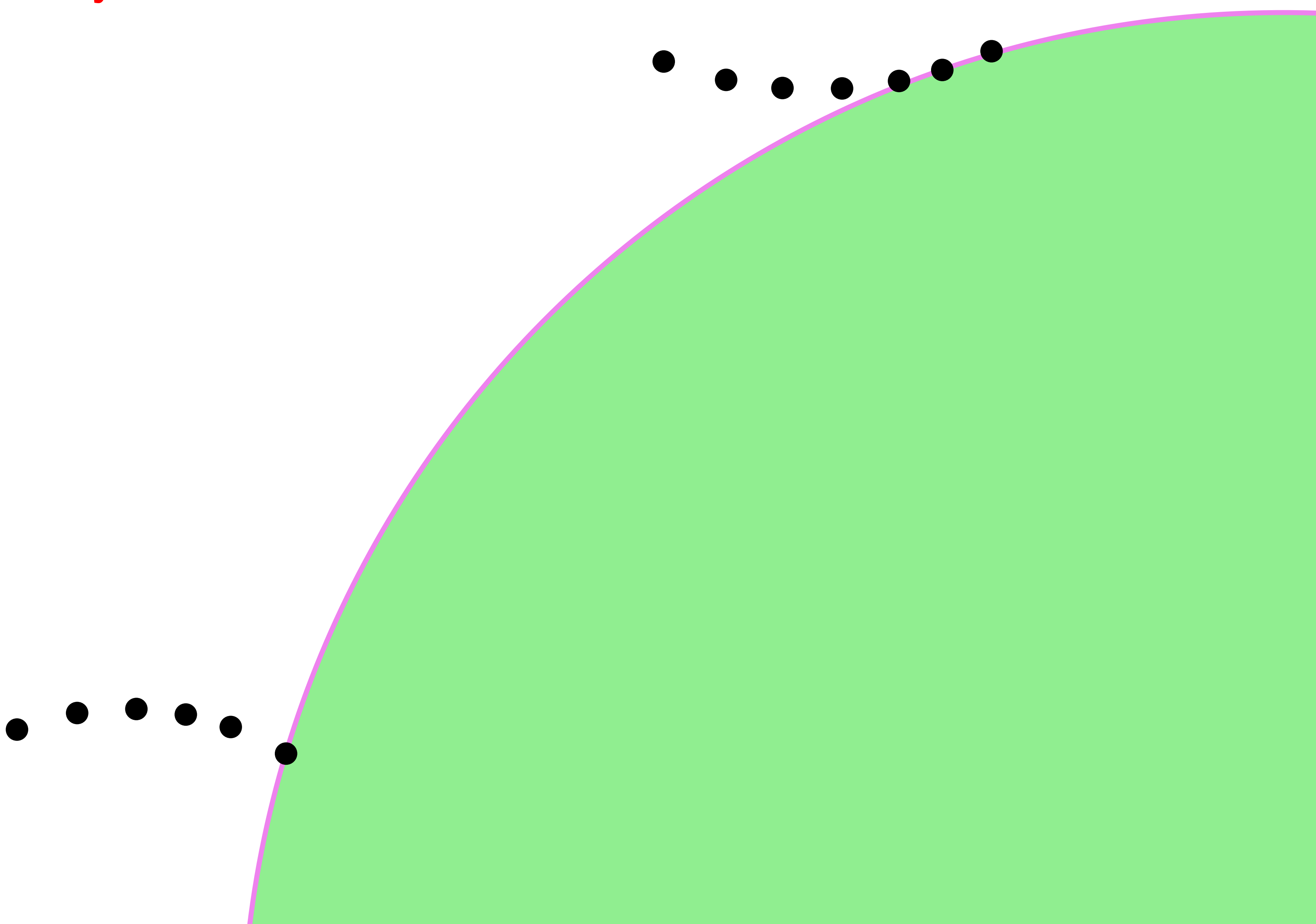
Delaunay Triangulation: Diagonal flipping

Complexity ?



Delaunay Triangulation: Diagonal flipping

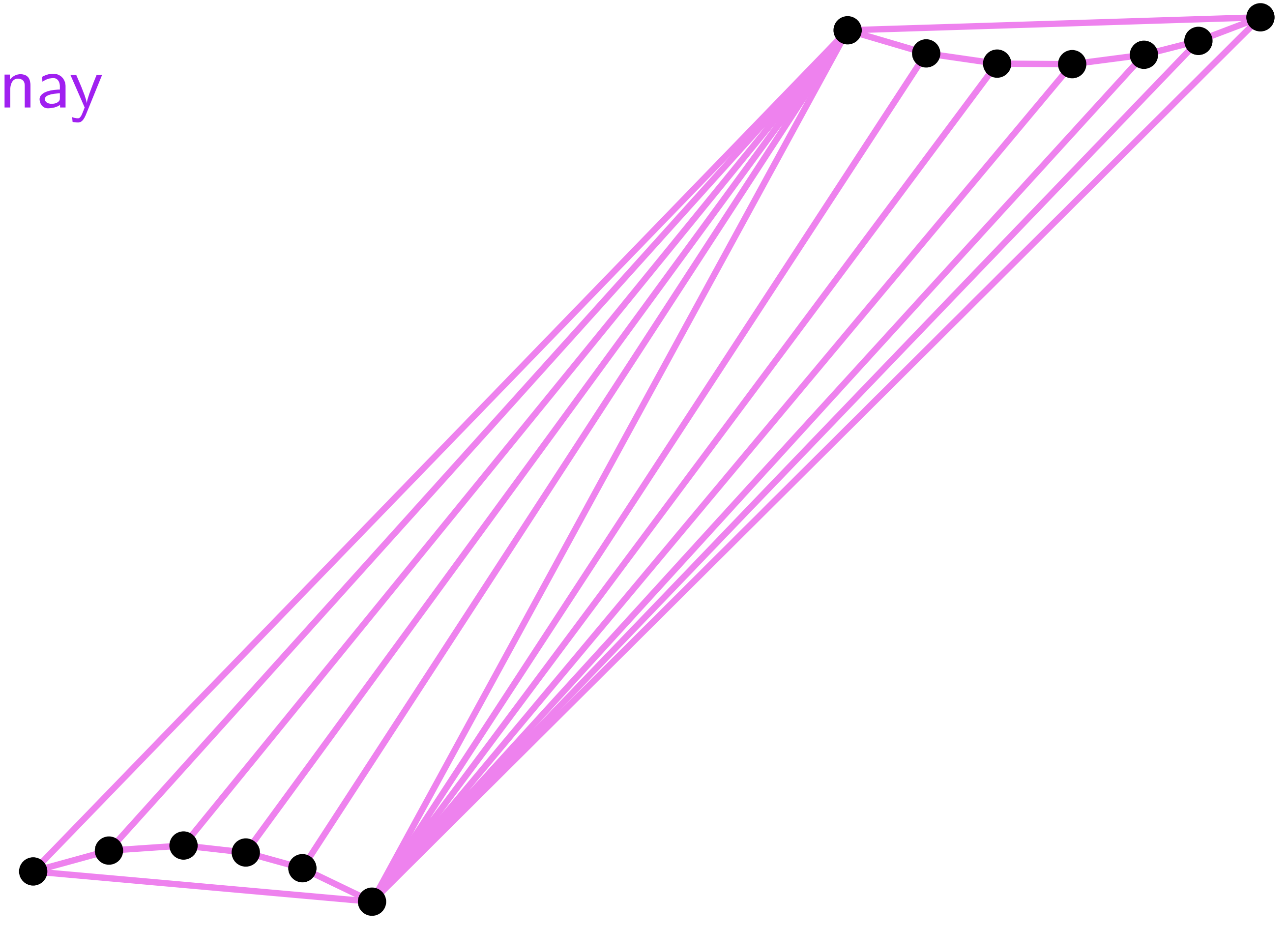
Complexity ?



Delaunay Triangulation: Diagonal flipping

Complexity ?

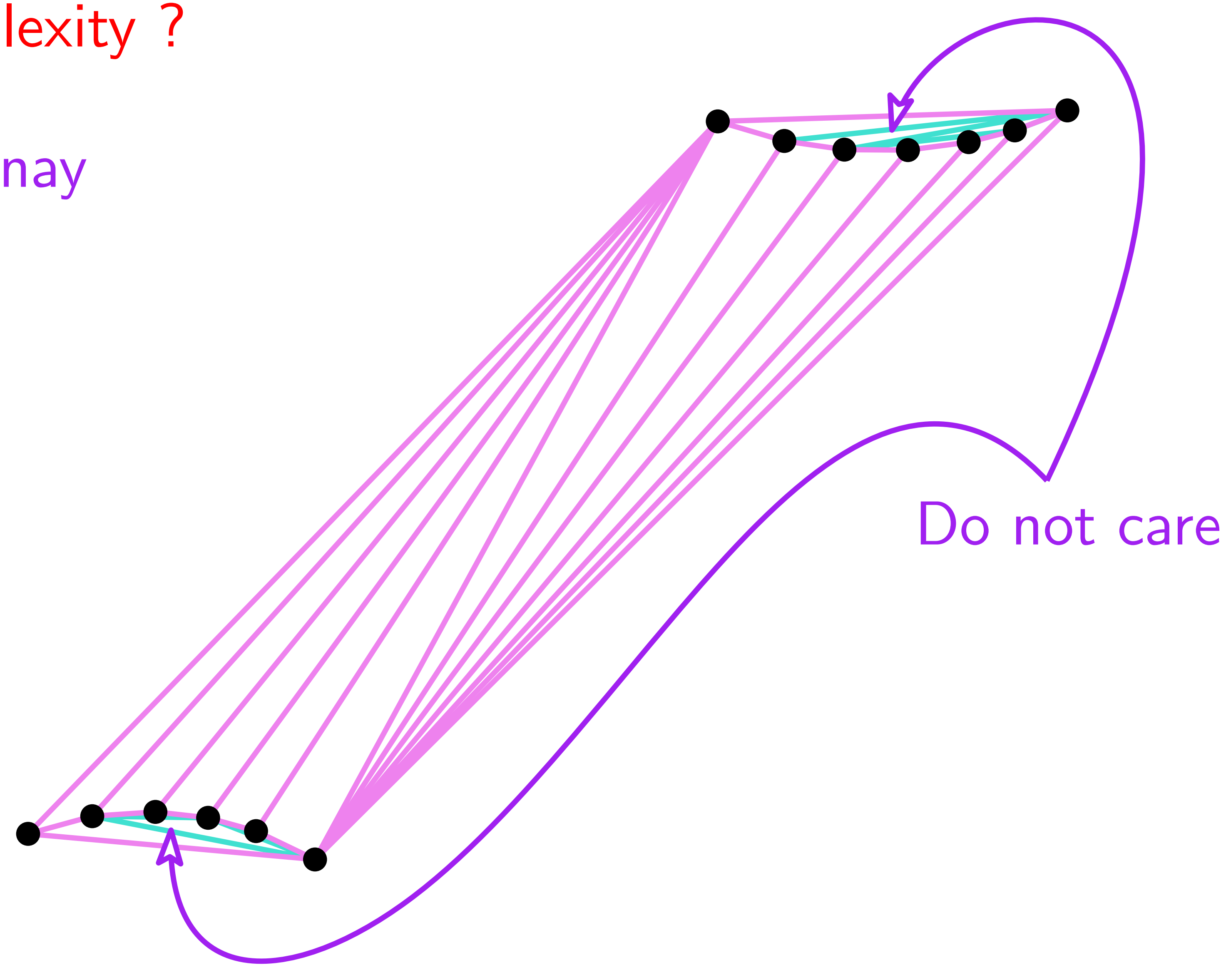
Delaunay



Delaunay Triangulation: Diagonal flipping

Complexity ?

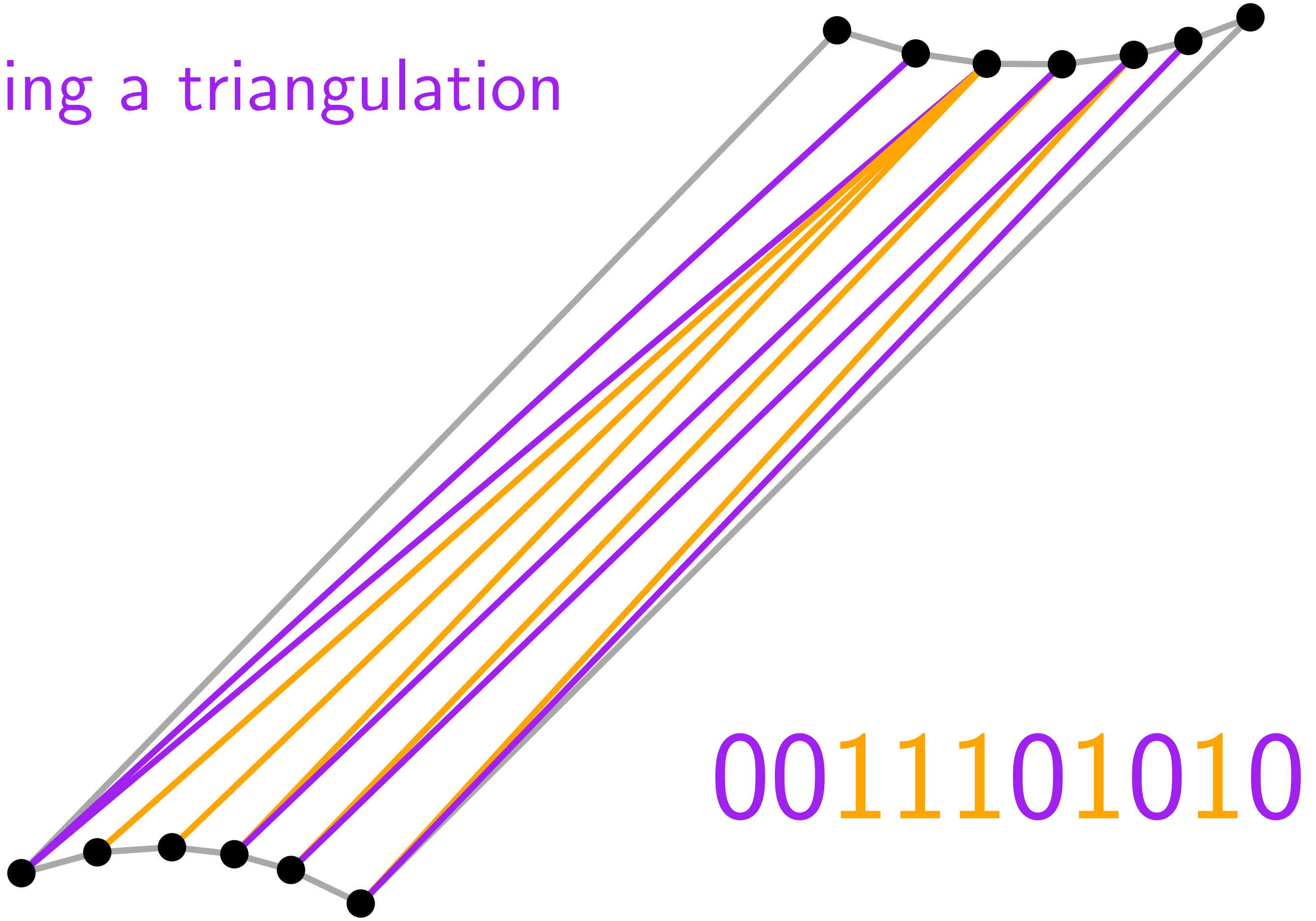
Delaunay



Delaunay Triangulation: Diagonal flipping

Complexity ?

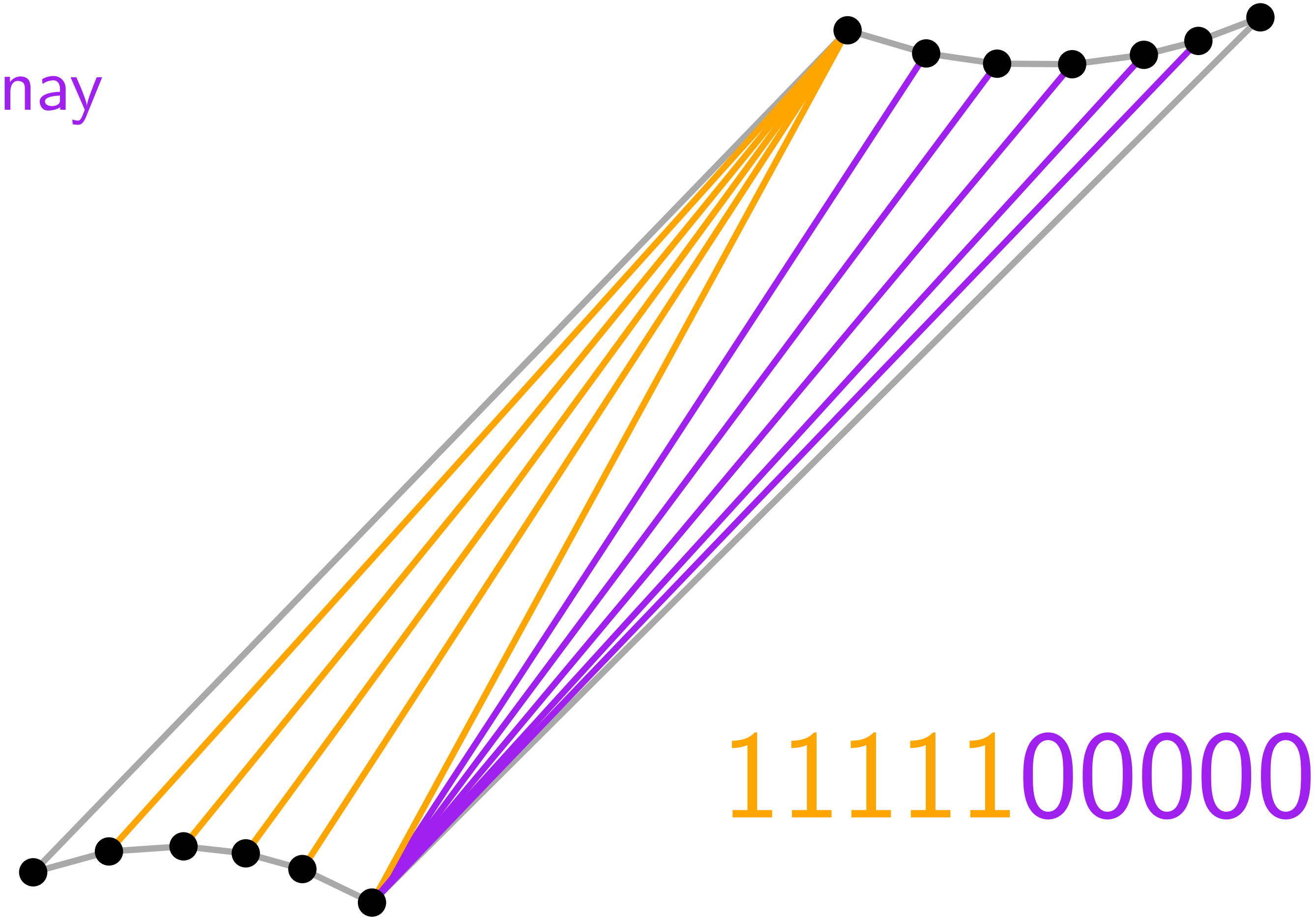
Encoding a triangulation



Delaunay Triangulation: Diagonal flipping

Complexity ?

Delaunay

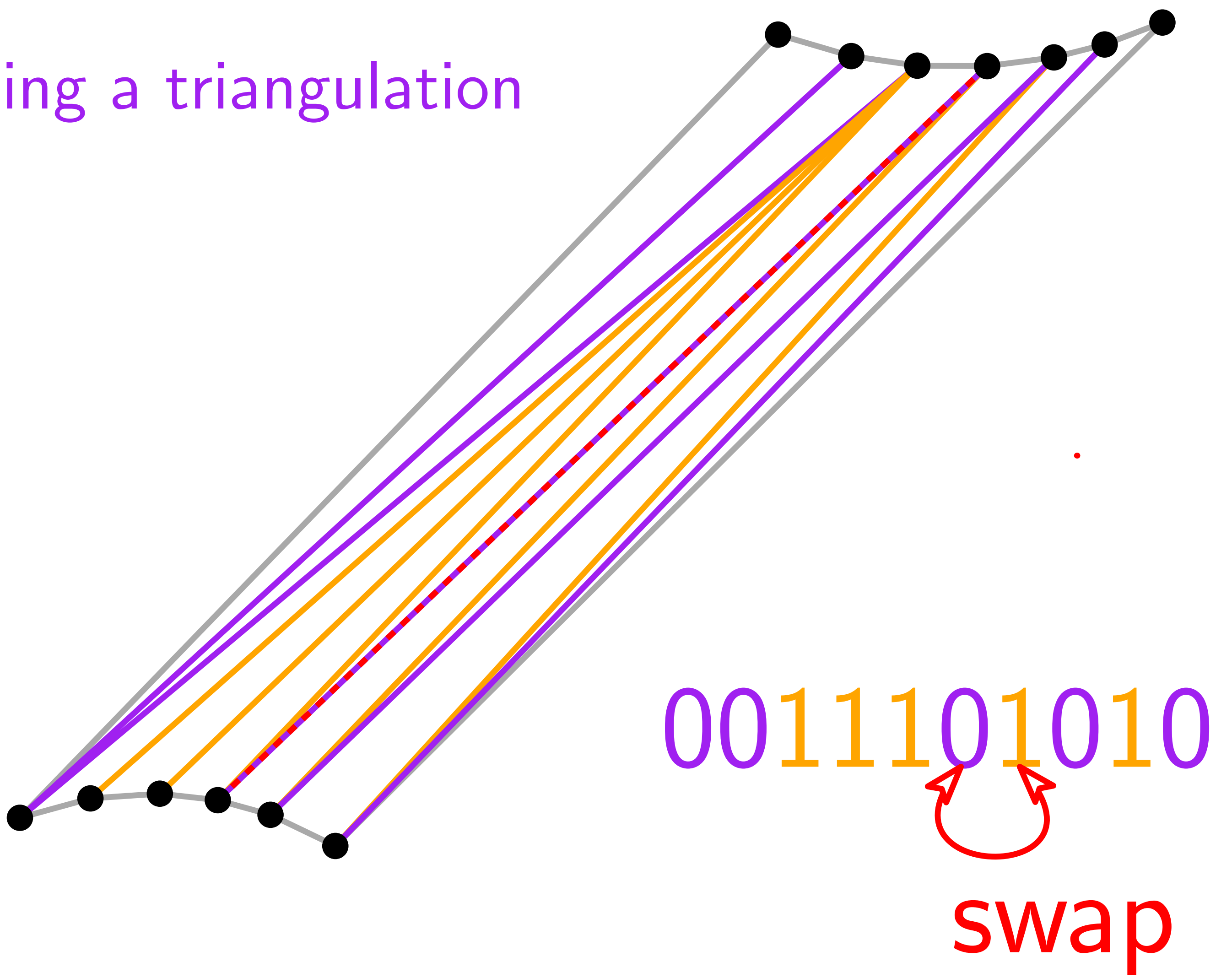


Delaunay Triangulation: Diagonal flipping

Complexity ?

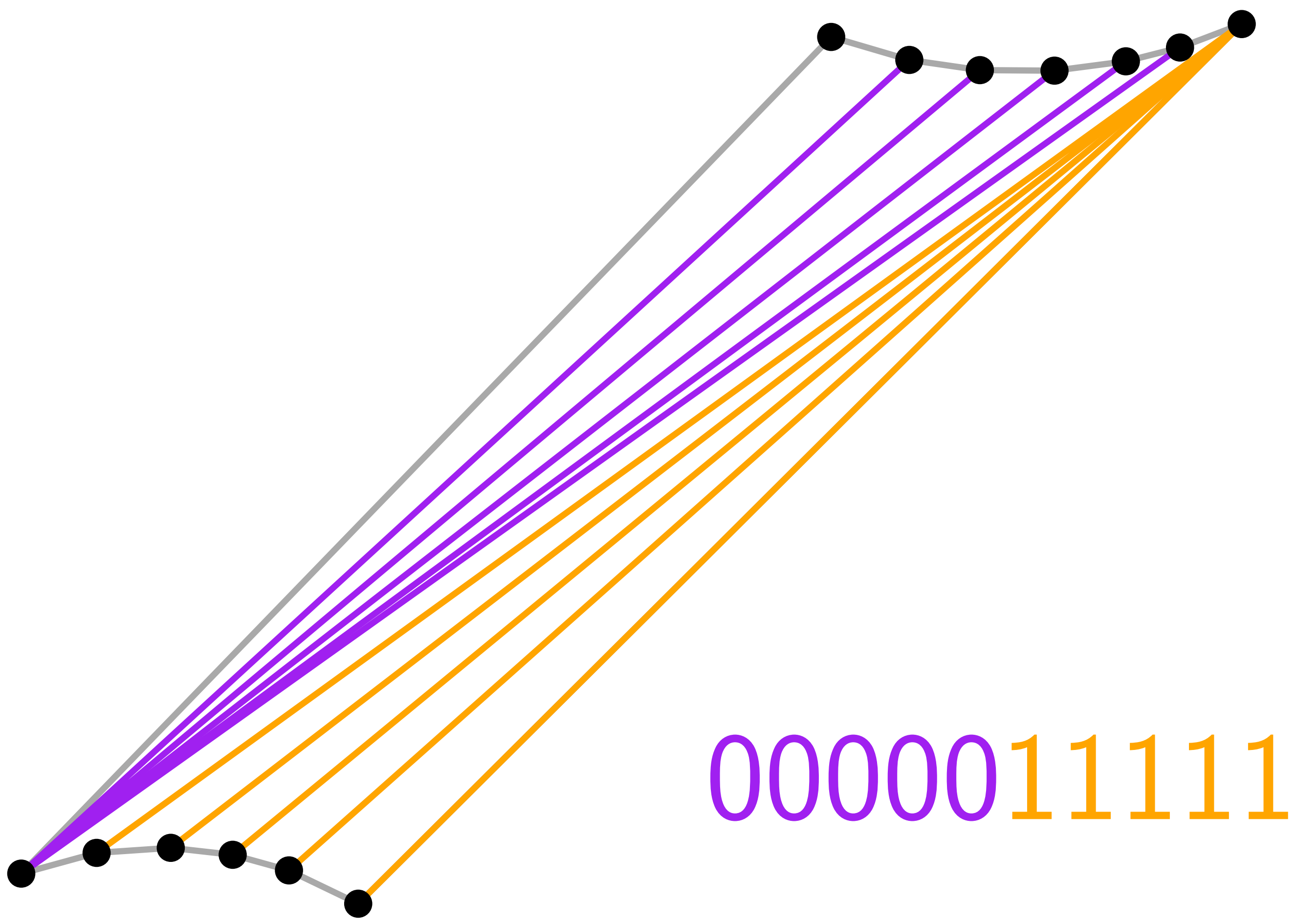
Encoding a triangulation

Flip



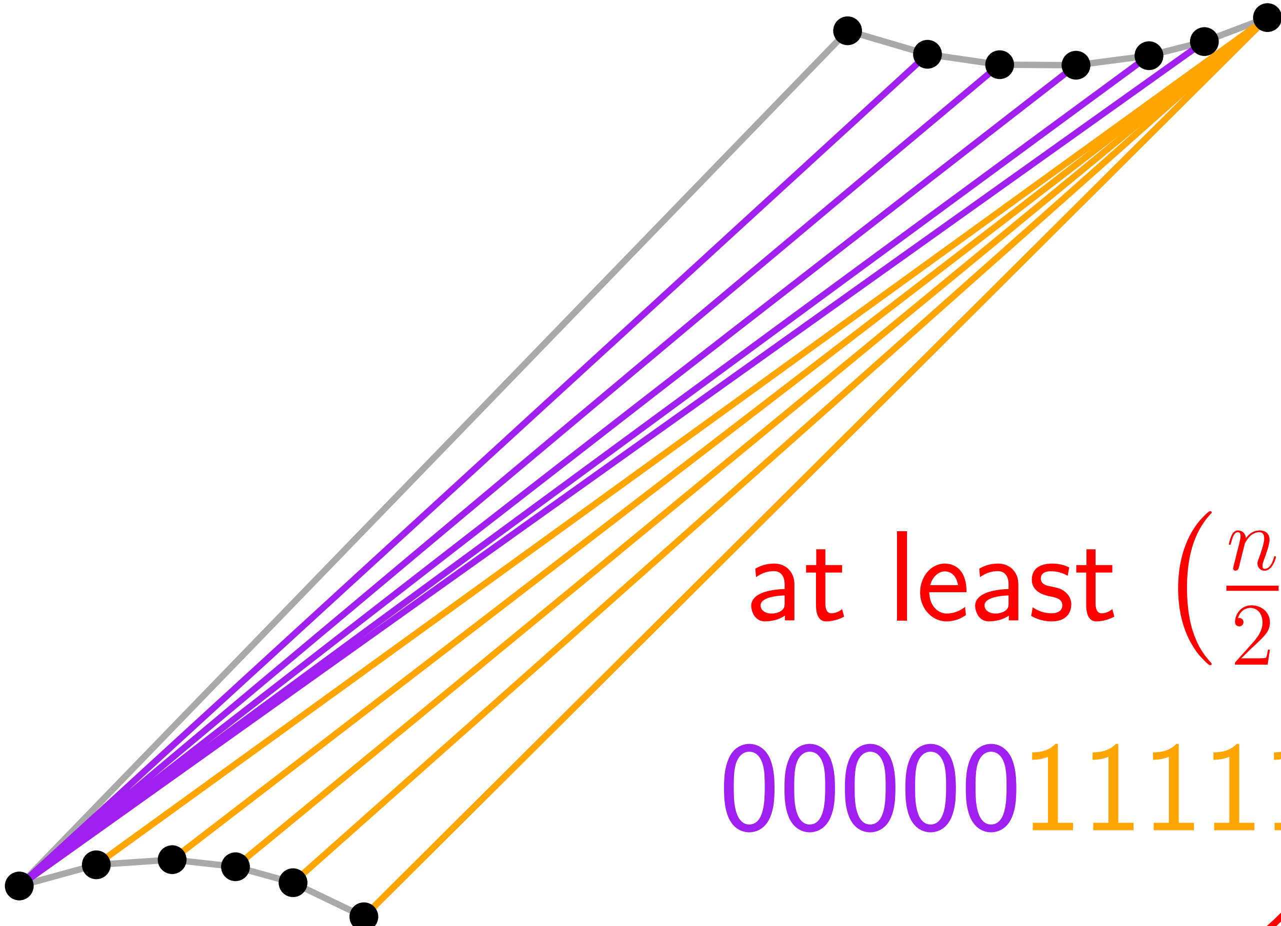
Delaunay Triangulation: Diagonal flipping

Complexity ?



Delaunay Triangulation: Diagonal flipping

Complexity ?

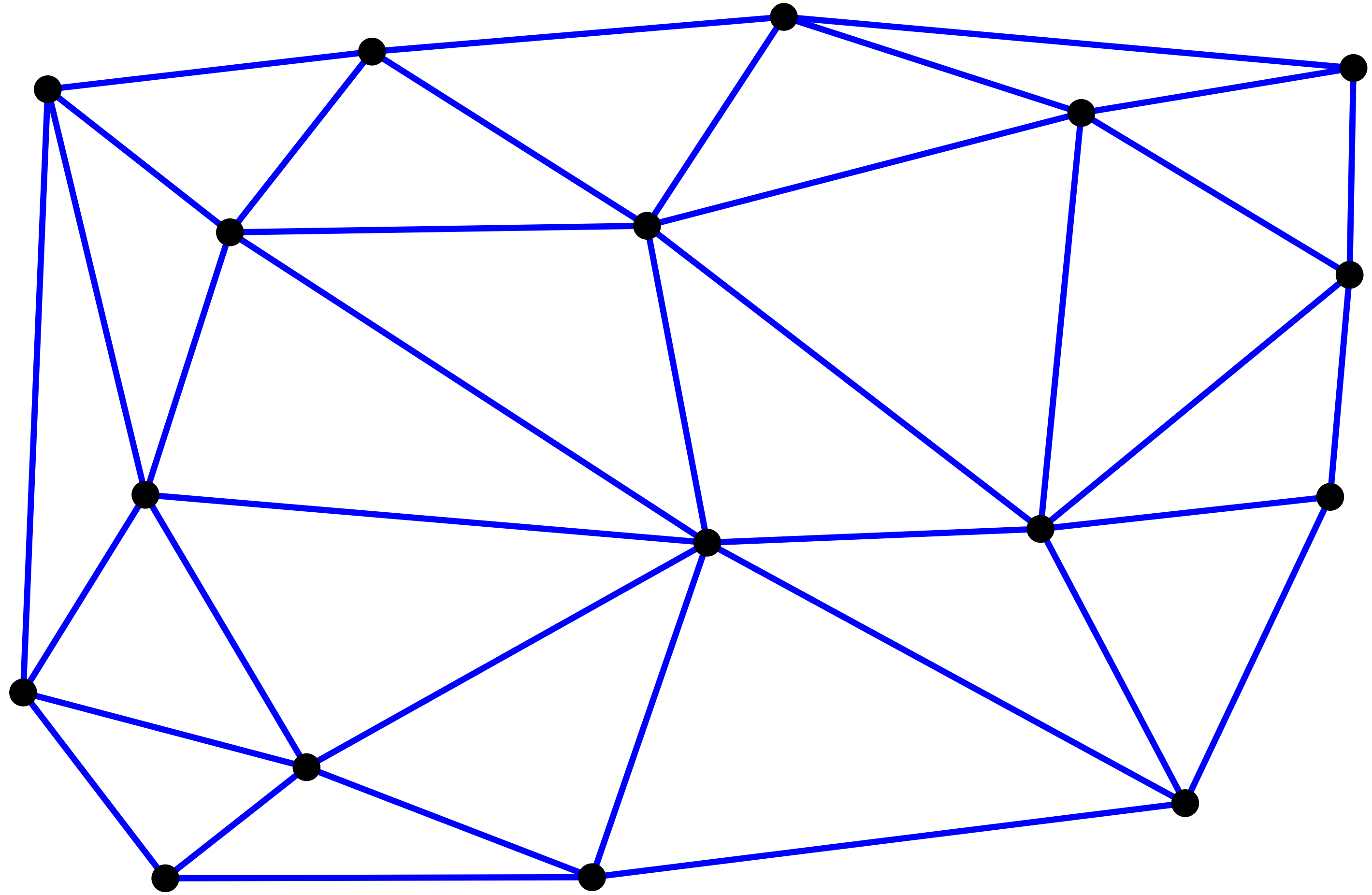


at least $\left(\frac{n}{2}\right)^2$ flips

0000011111

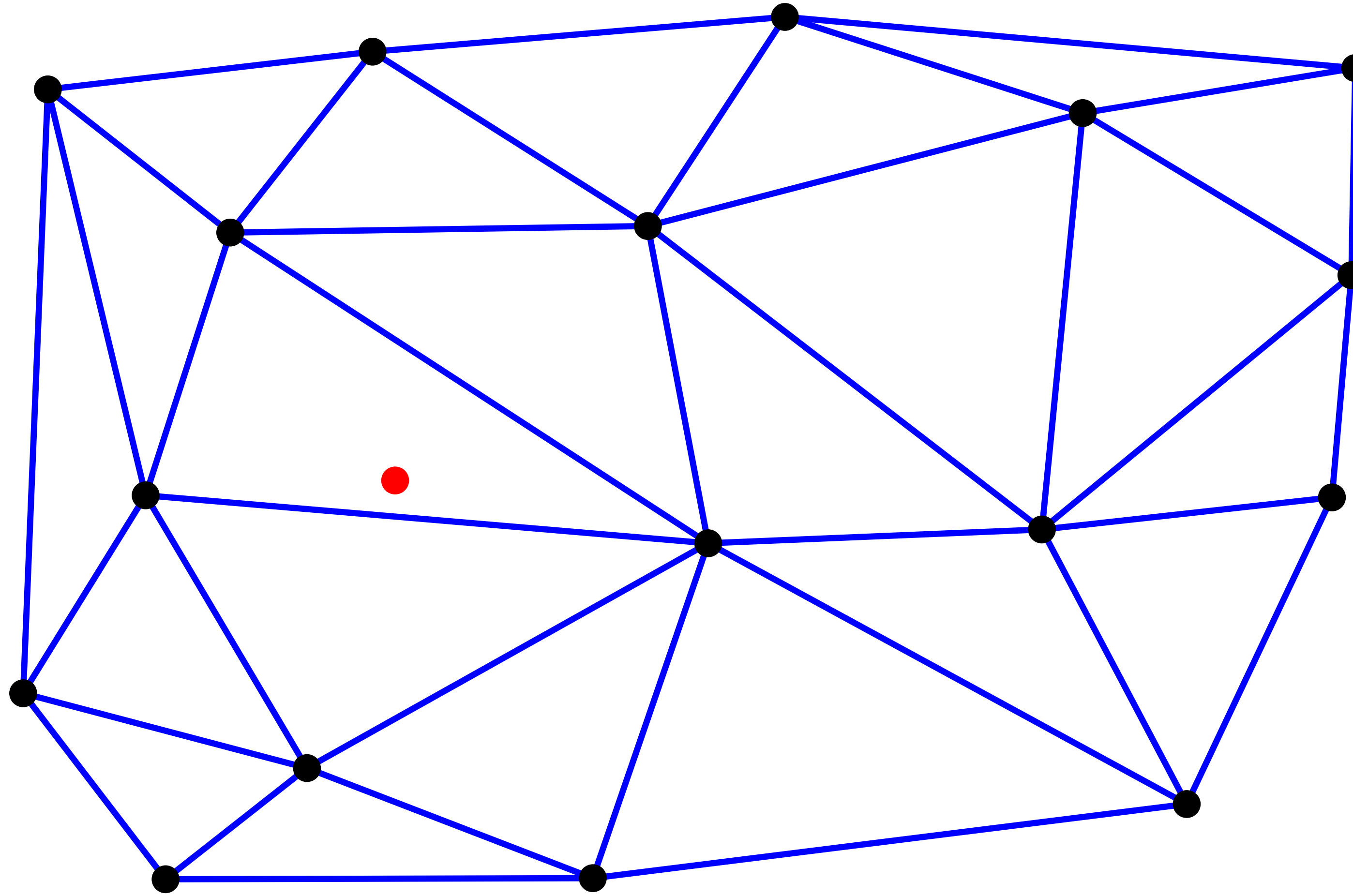
Delaunay 1111100000

Delaunay Triangulation: incremental algorithm



Delaunay Triangulation: incremental algorithm

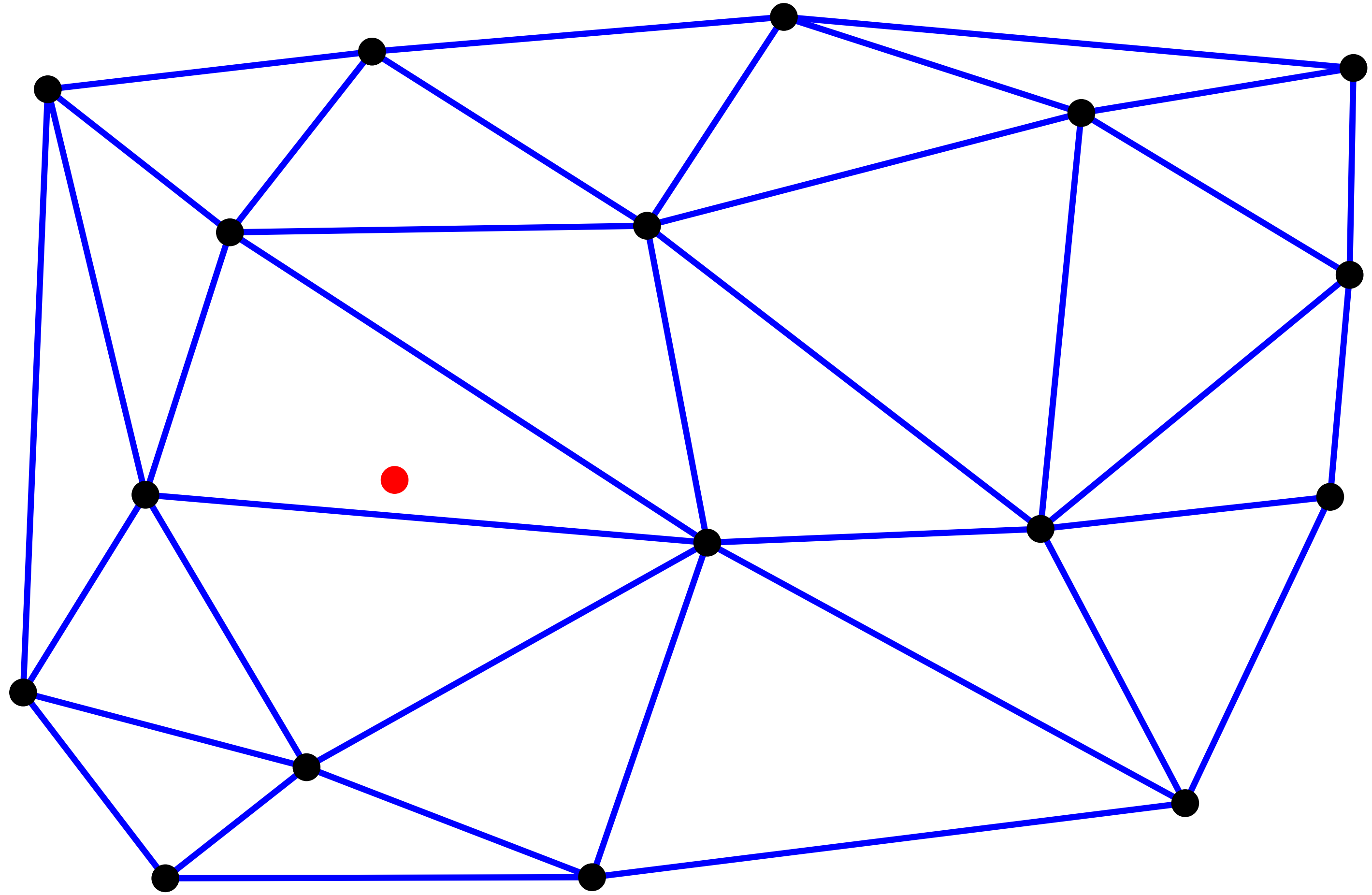
New point



Delaunay Triangulation: incremental algorithm

New point

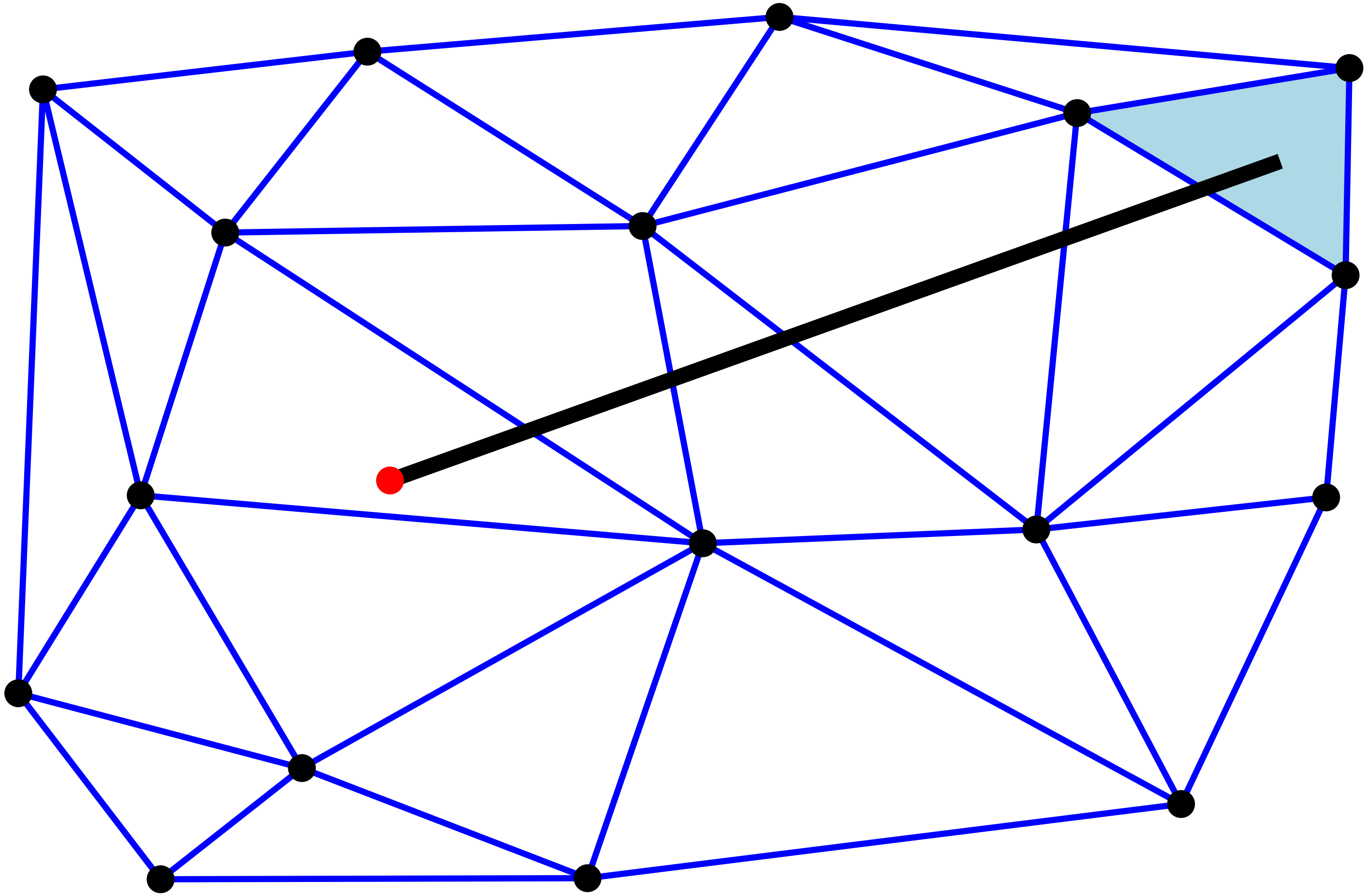
Locate



Delaunay Triangulation: incremental algorithm

New point

Locate

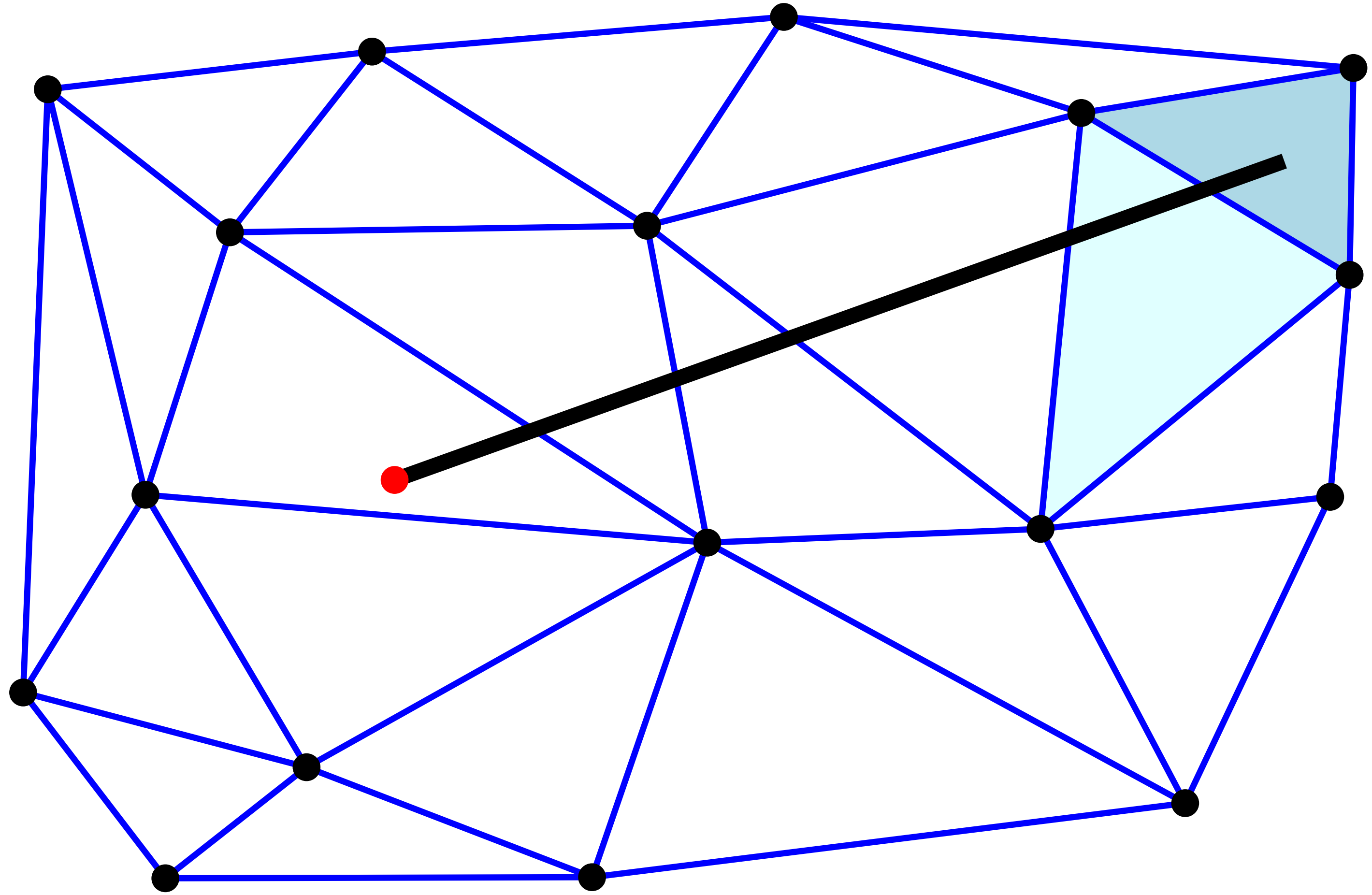


e.g.: straight walk

Delaunay Triangulation: incremental algorithm

New point

Locate

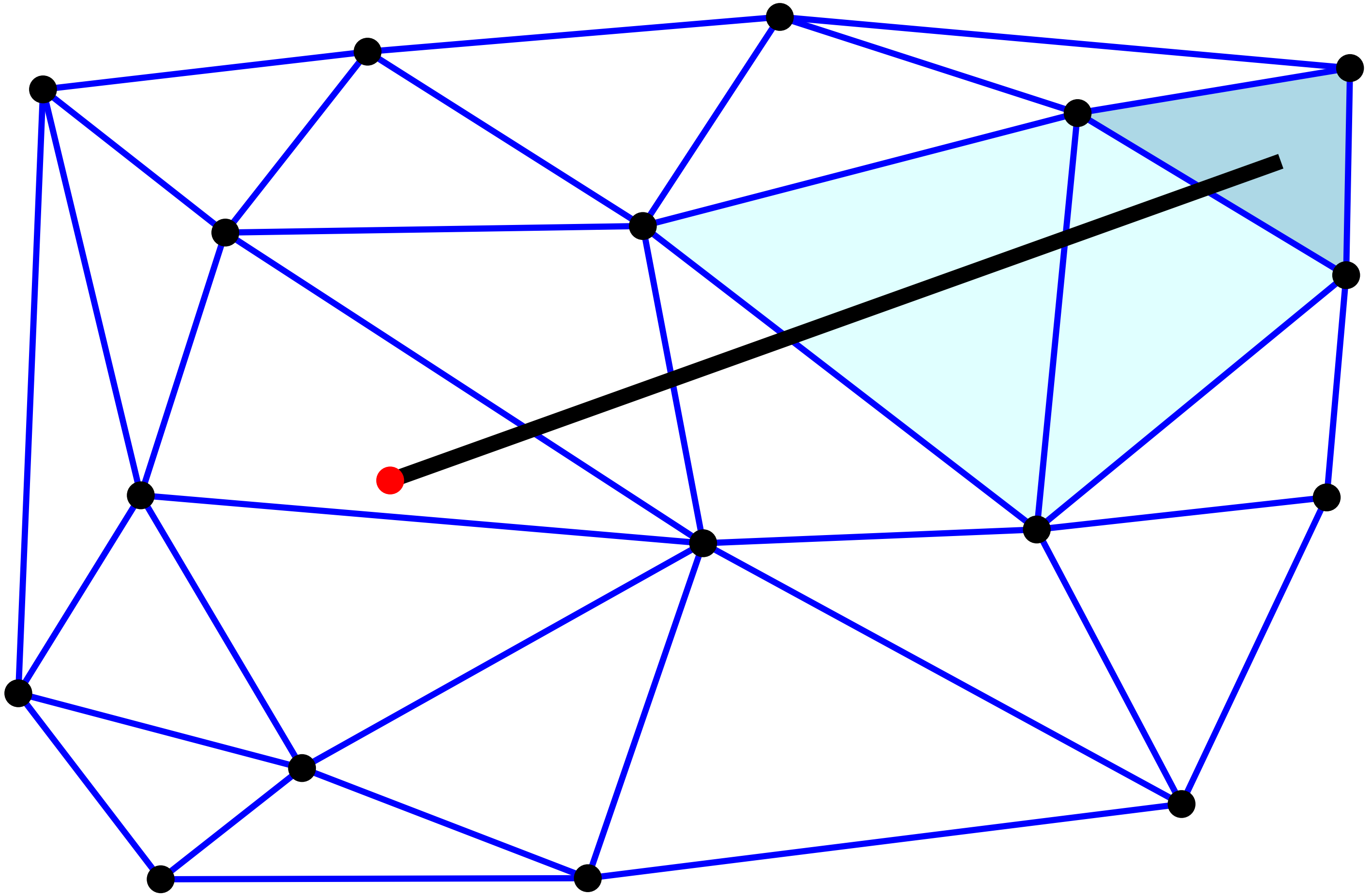


e.g.: straight walk

Delaunay Triangulation: incremental algorithm

New point

Locate

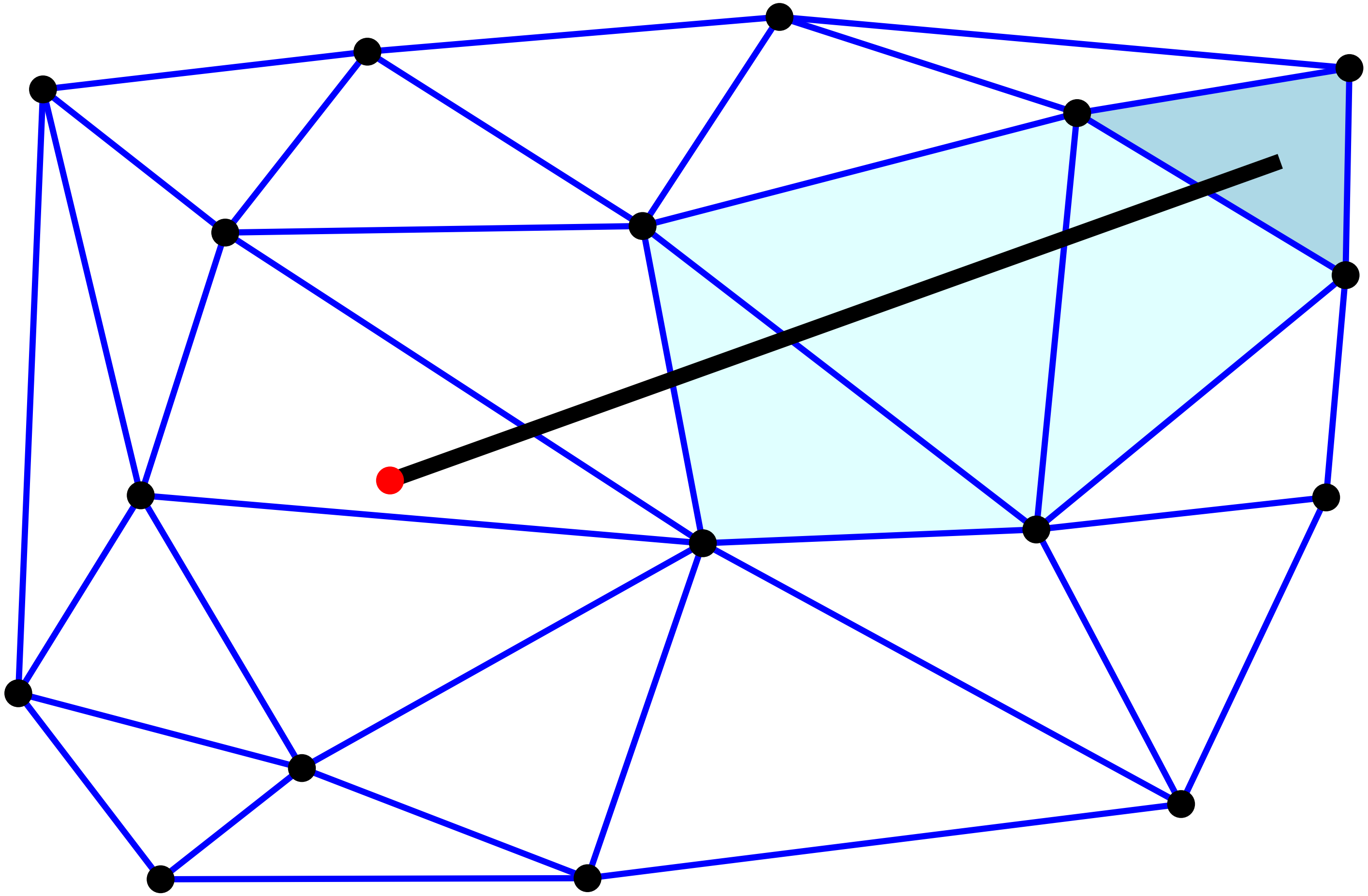


e.g.: straight walk

Delaunay Triangulation: incremental algorithm

New point

Locate

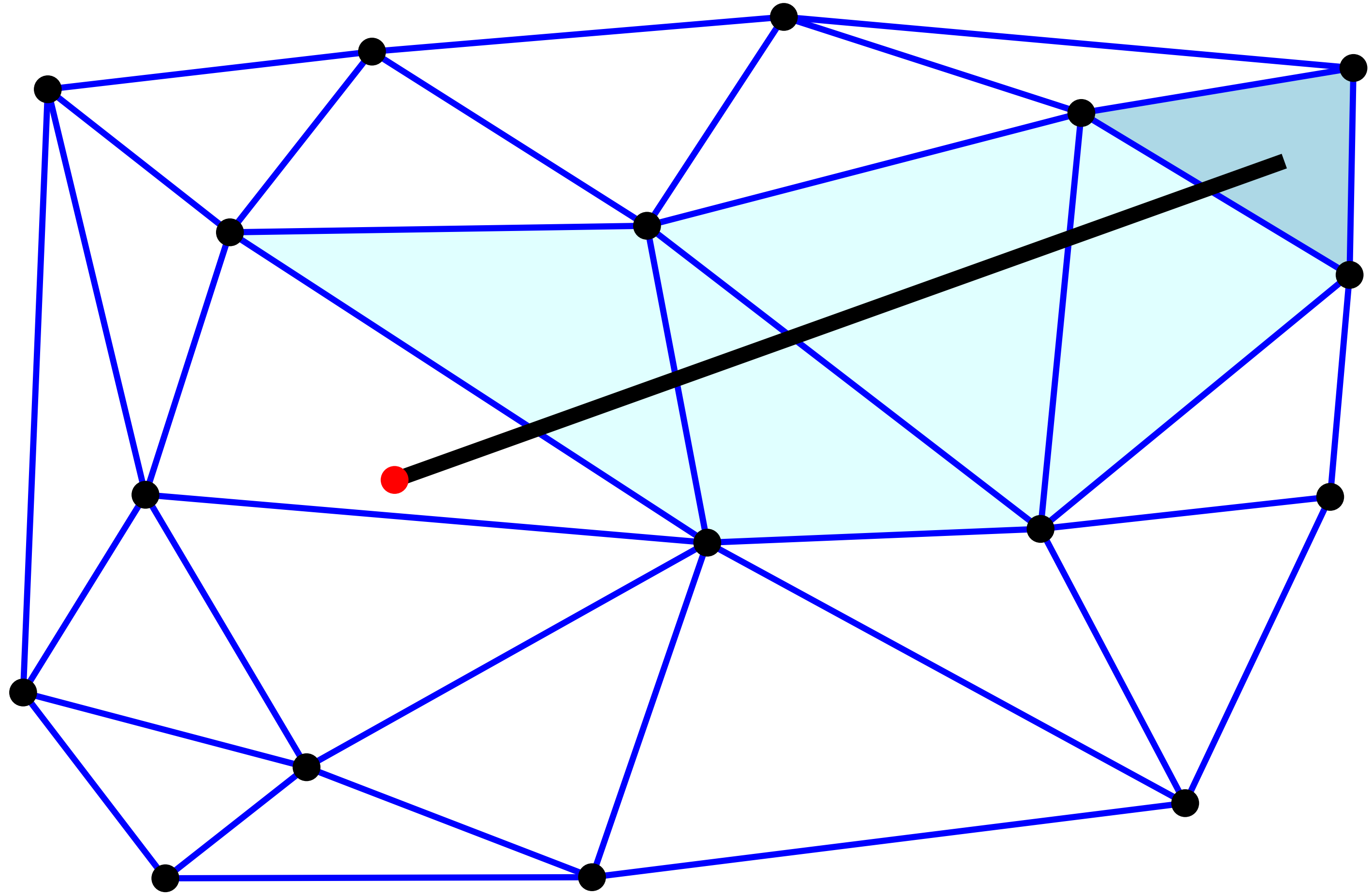


e.g.: straight walk

Delaunay Triangulation: incremental algorithm

New point

Locate

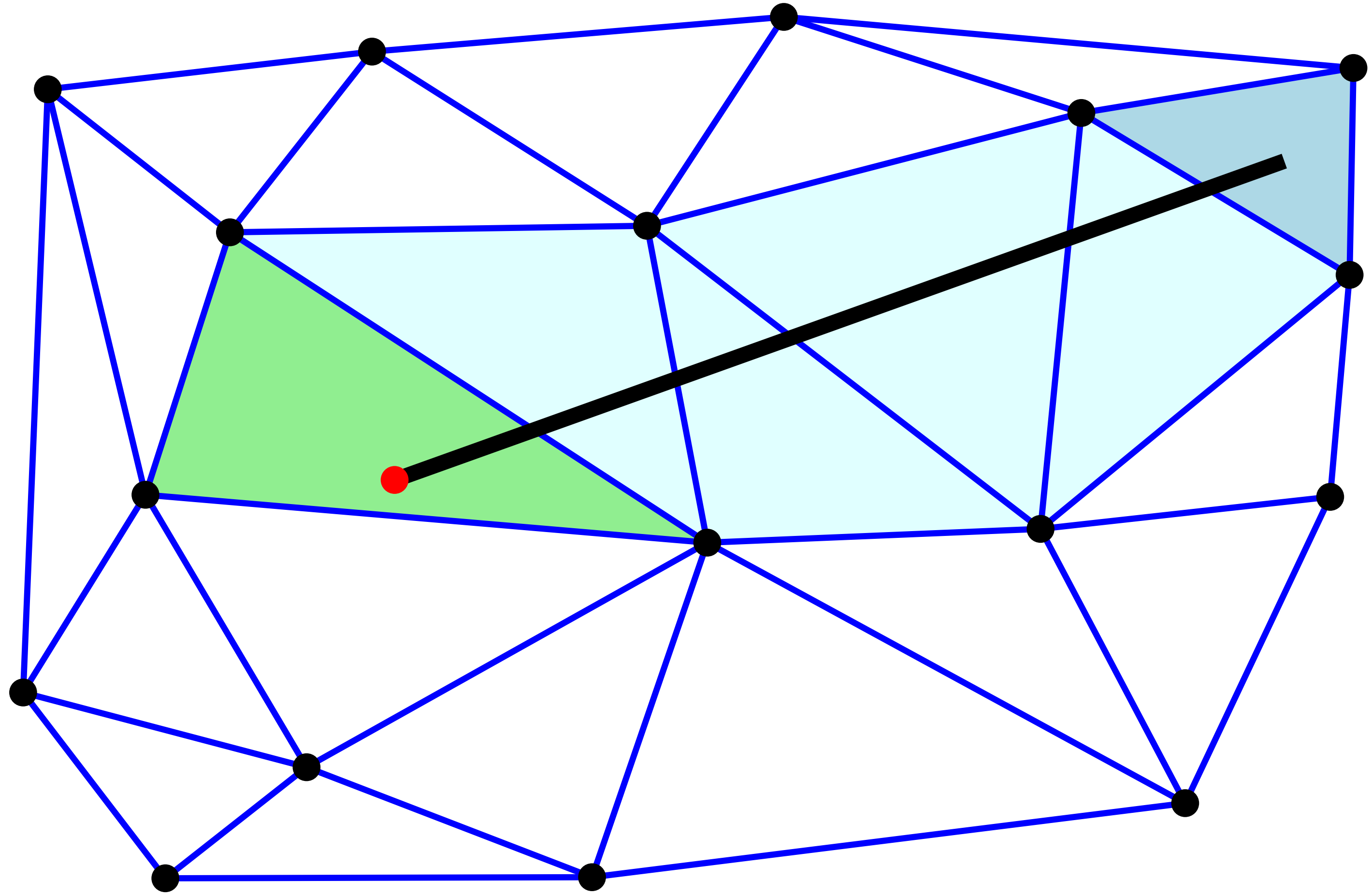


e.g.: straight walk

Delaunay Triangulation: incremental algorithm

New point

Locate

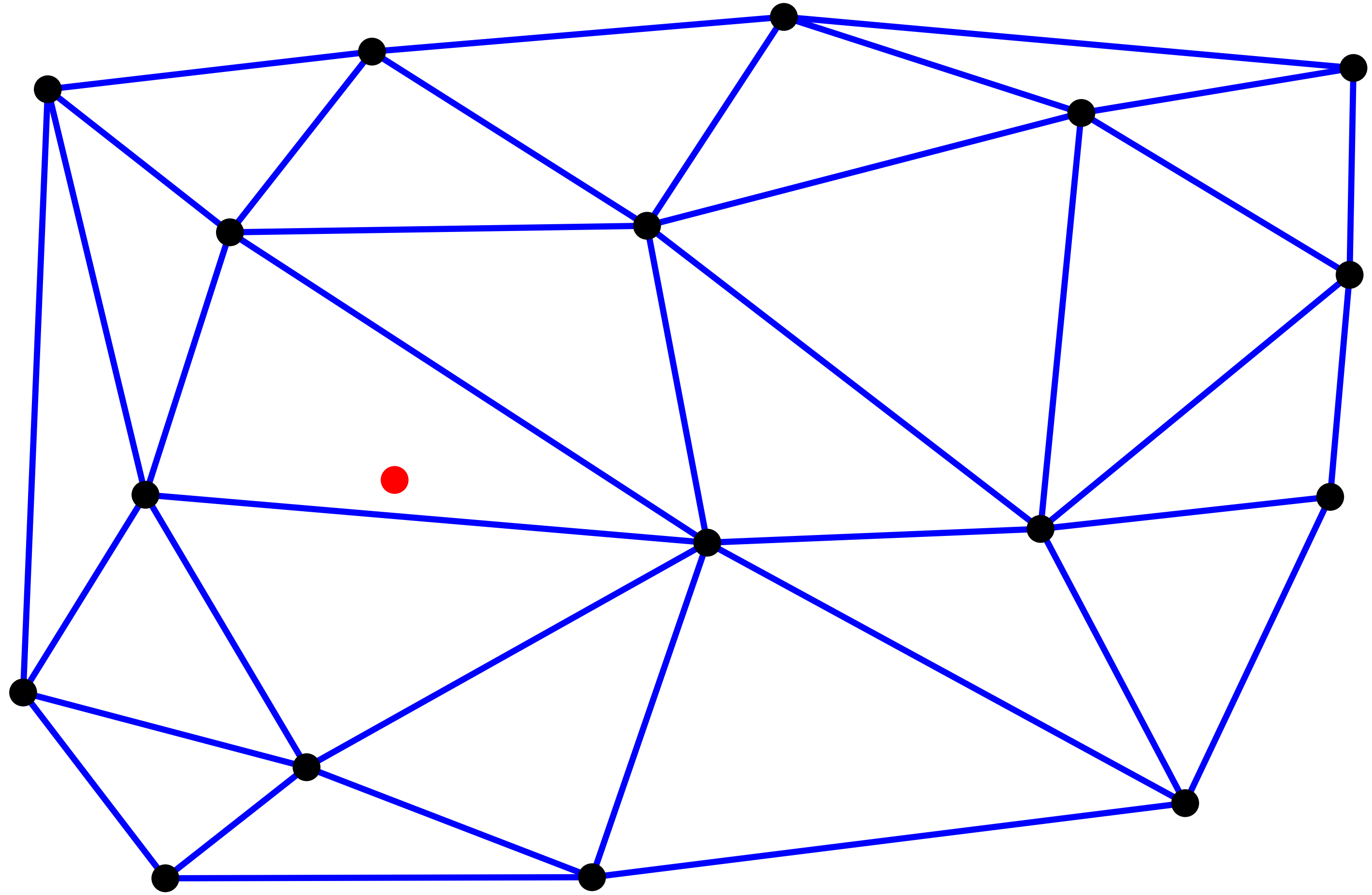


e.g.: straight walk

Delaunay Triangulation: incremental algorithm

New point

Locate

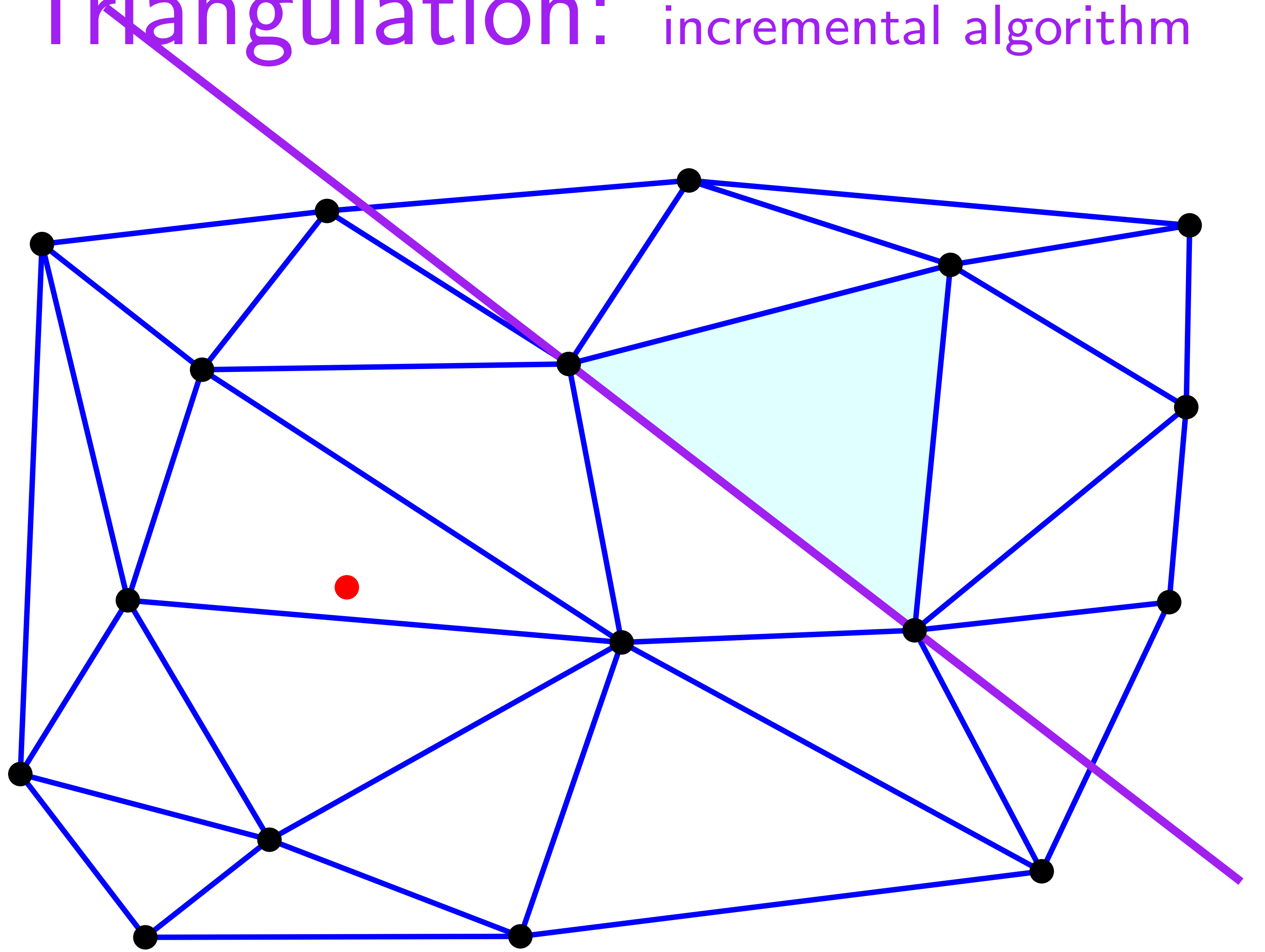


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

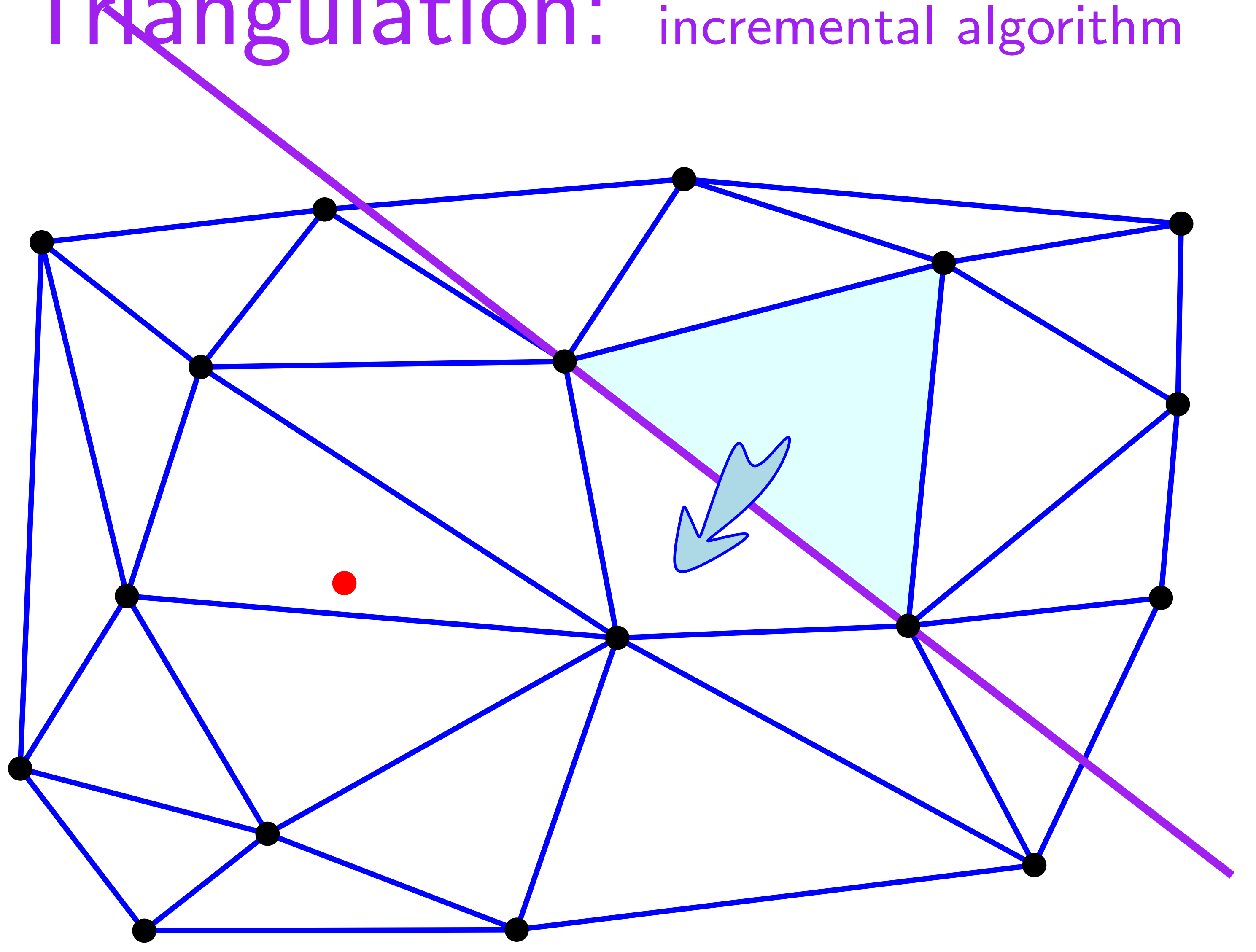


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

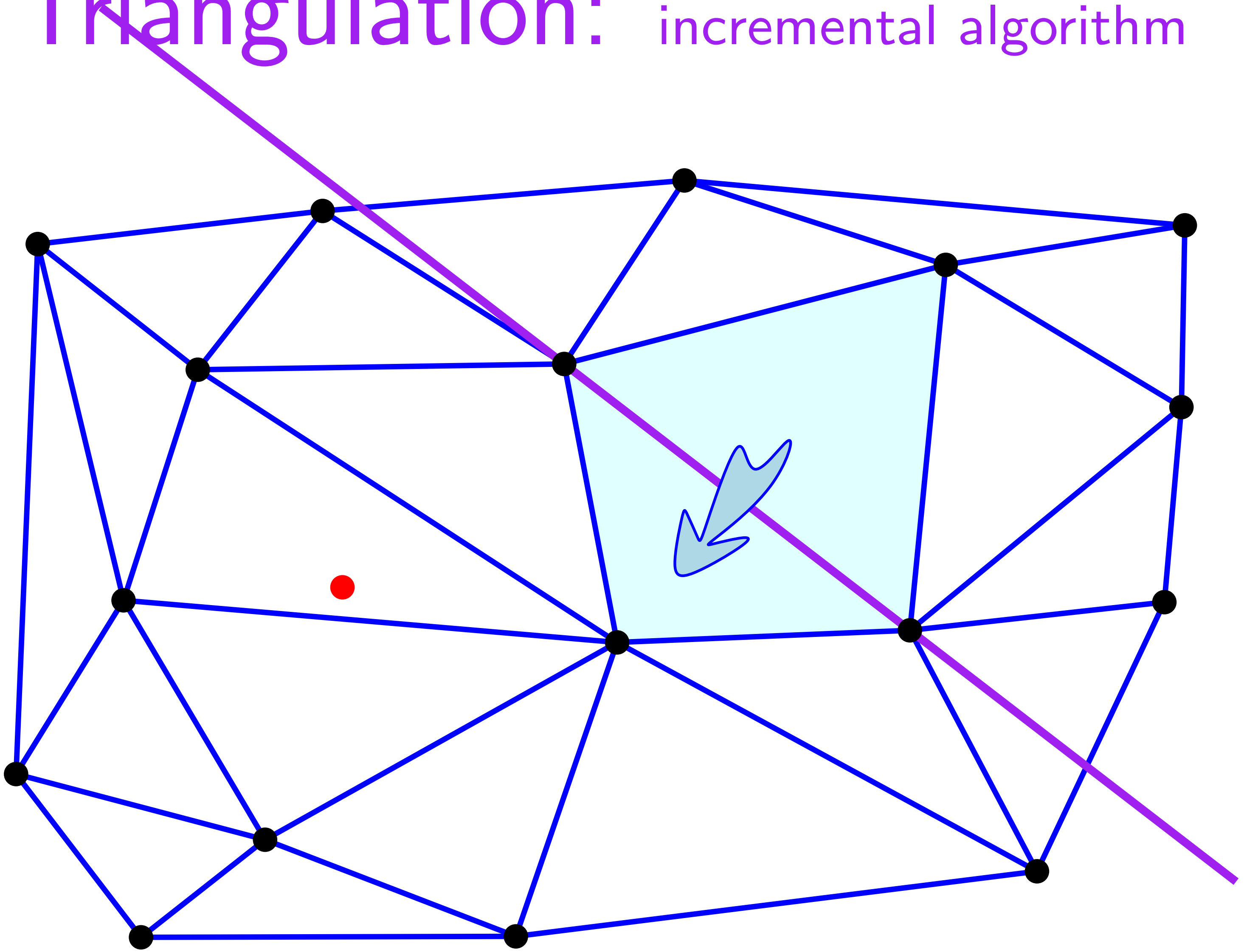


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

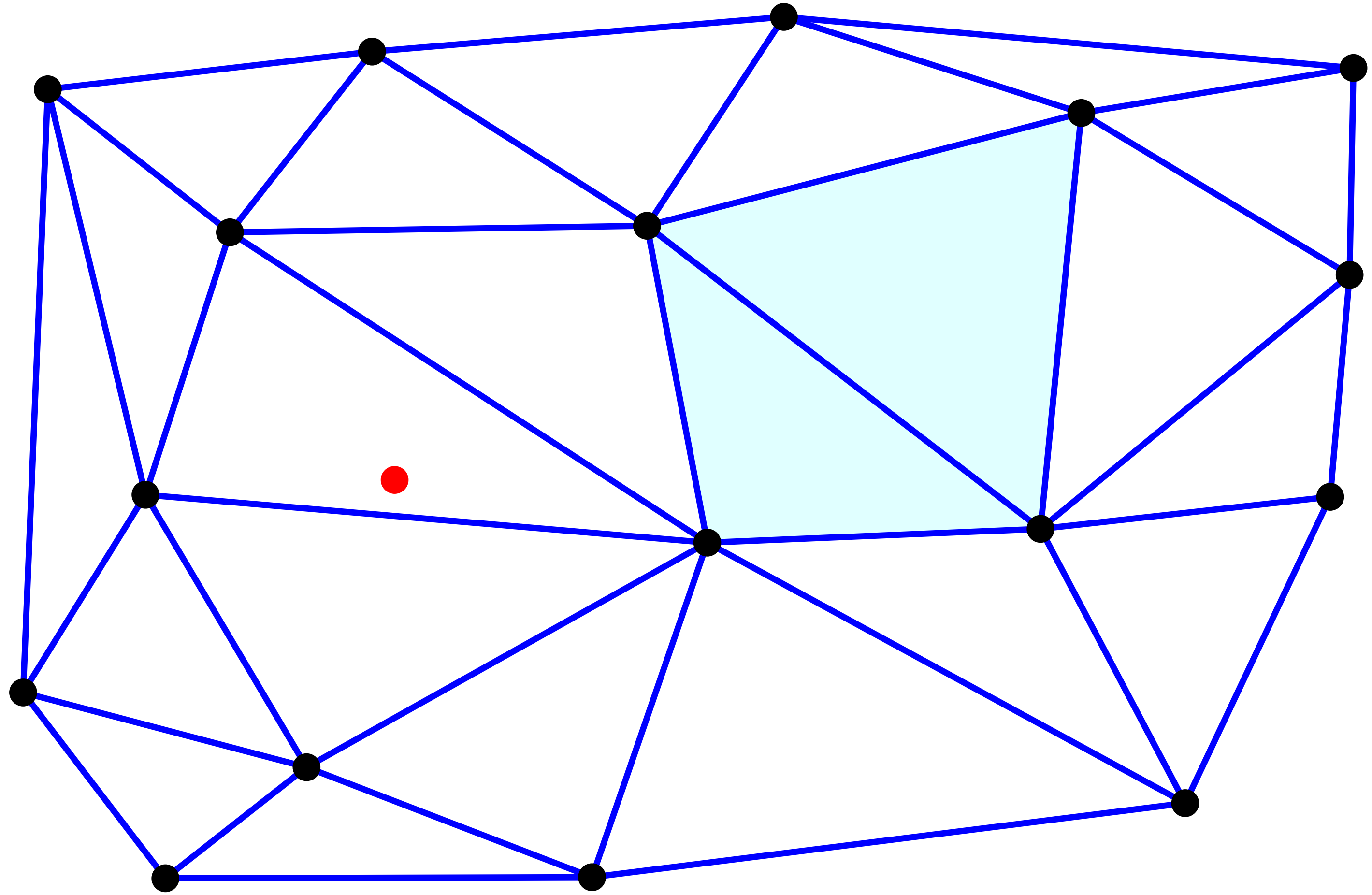


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

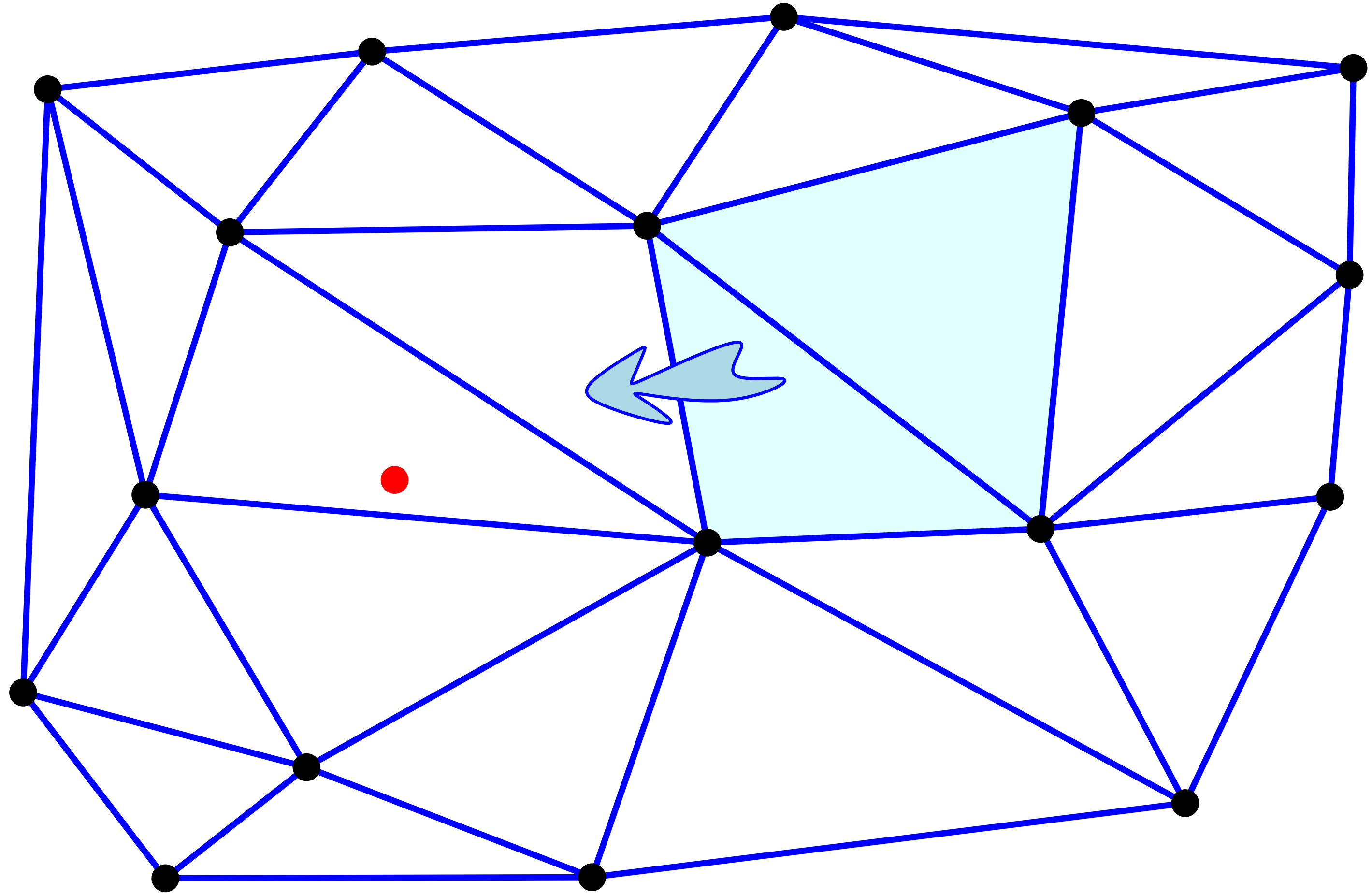


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

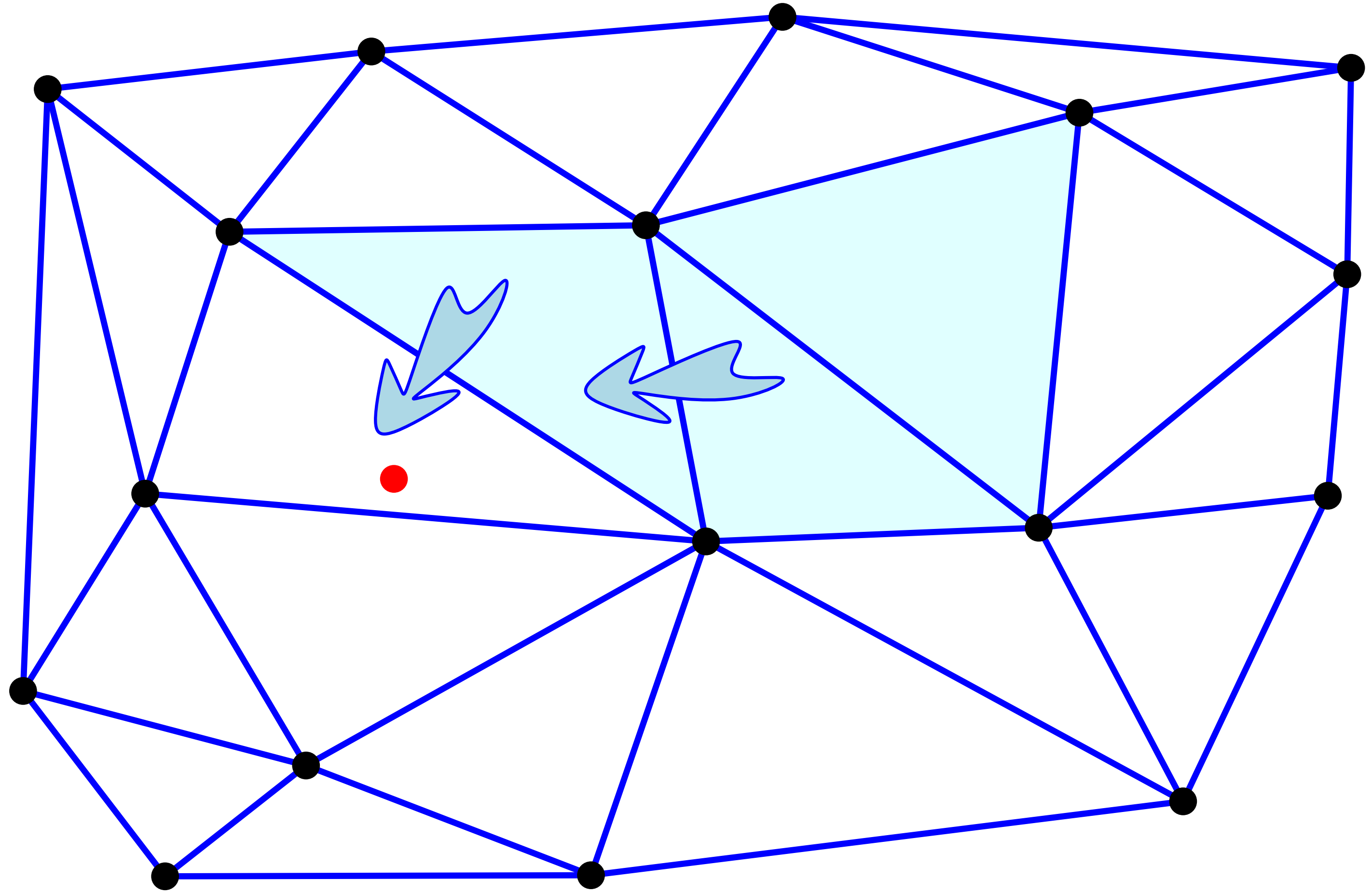


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

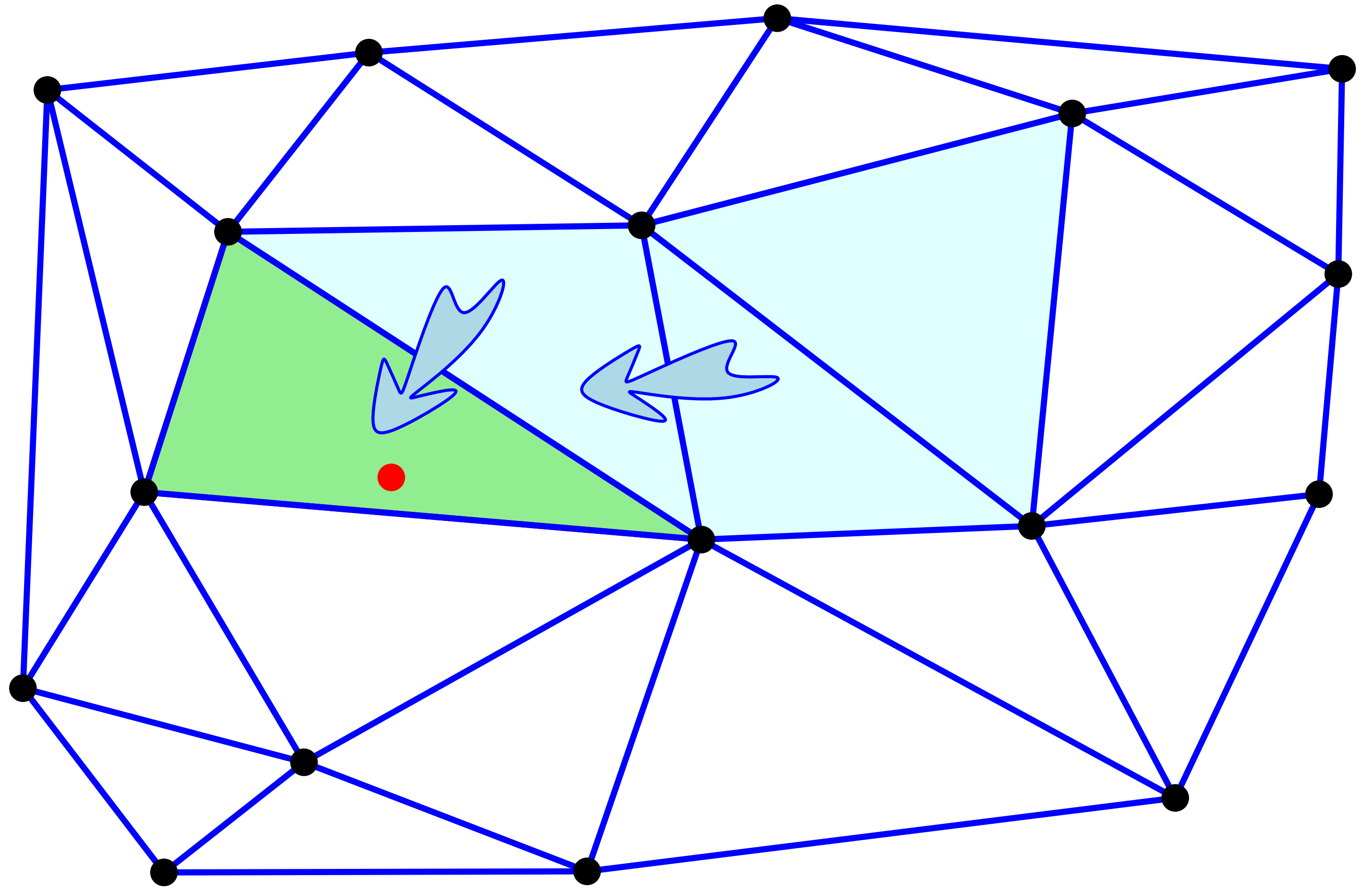


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

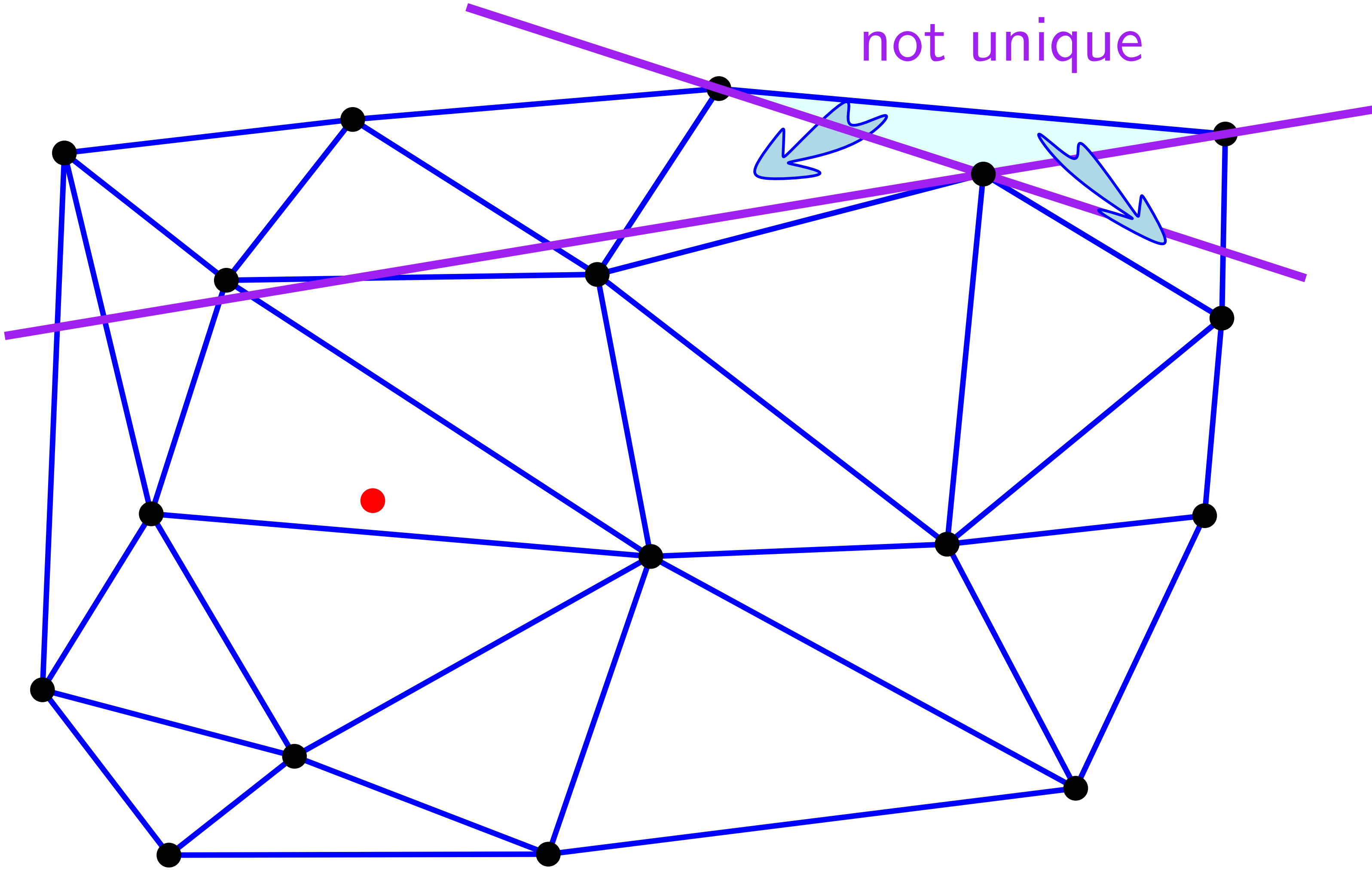


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate

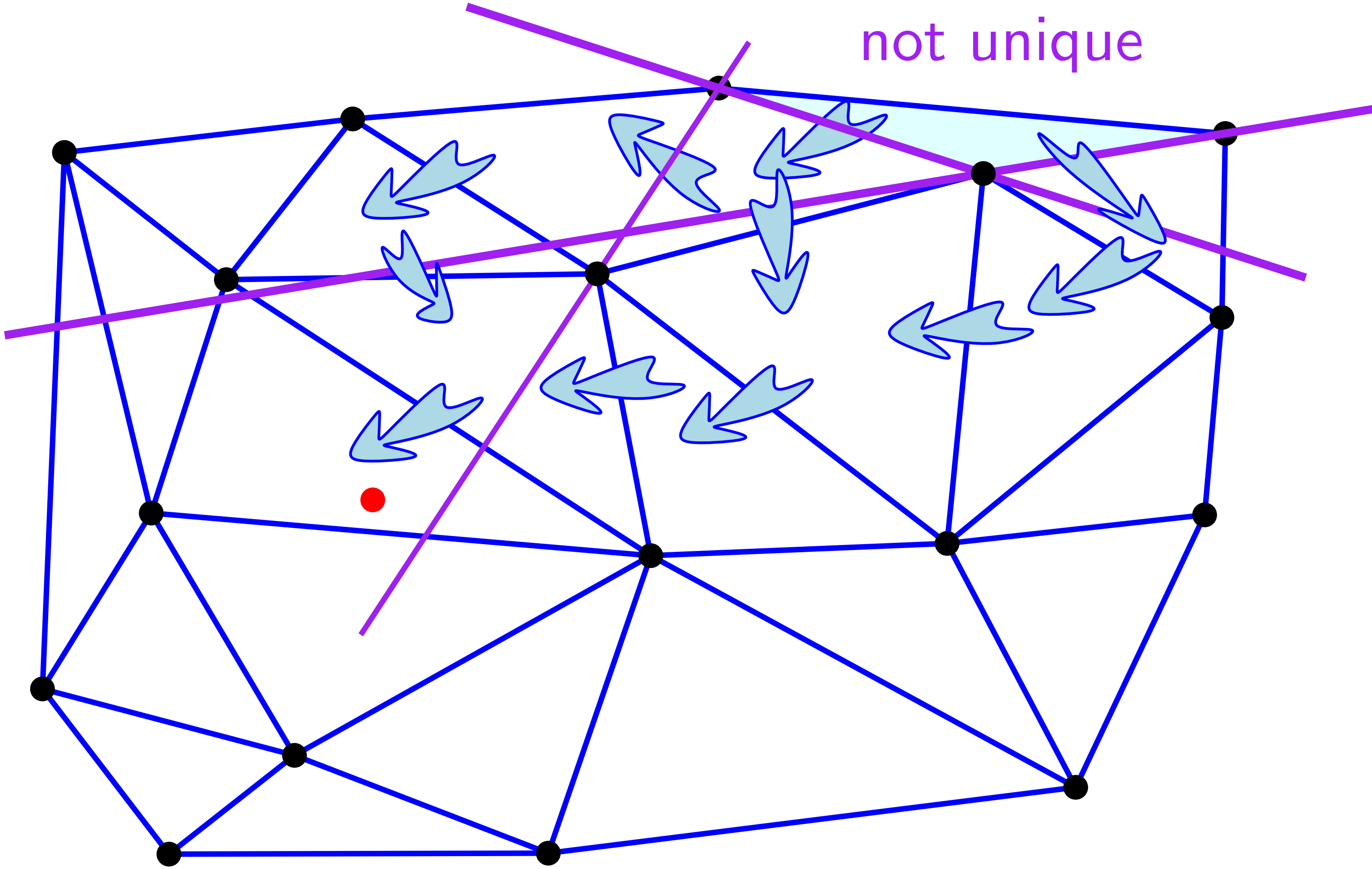


e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

New point

Locate



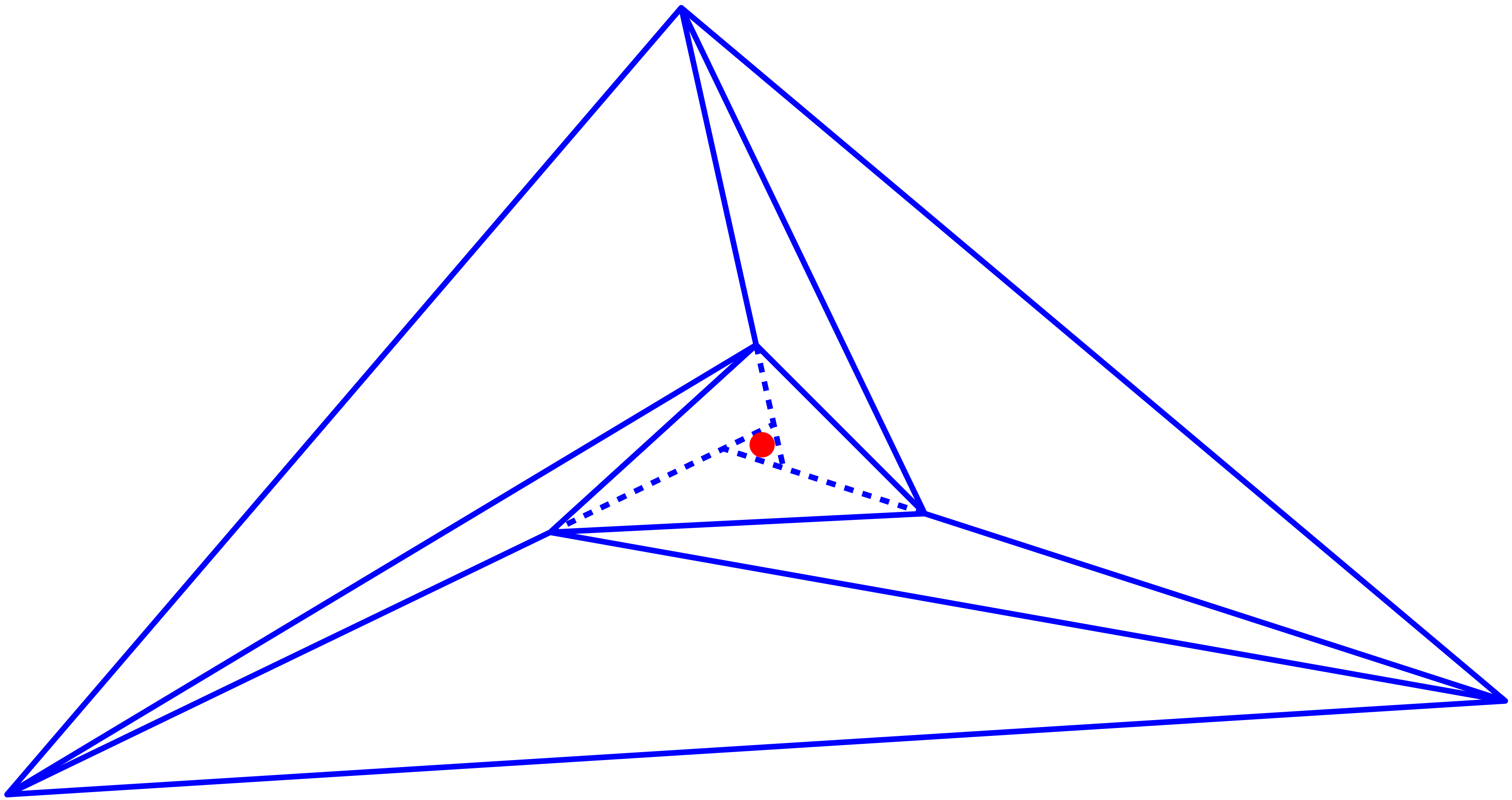
e.g.: visibility walk

Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?

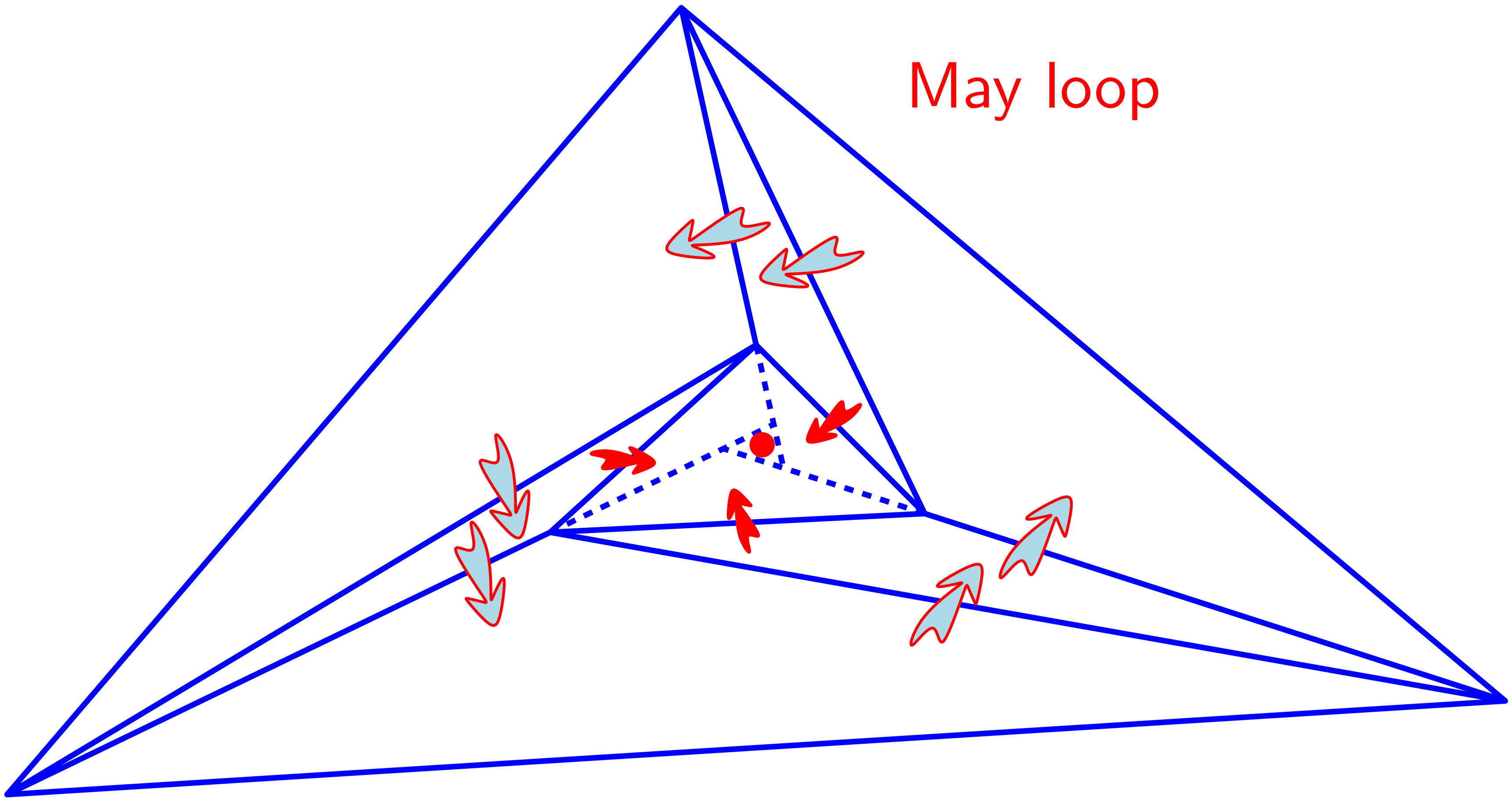
Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?



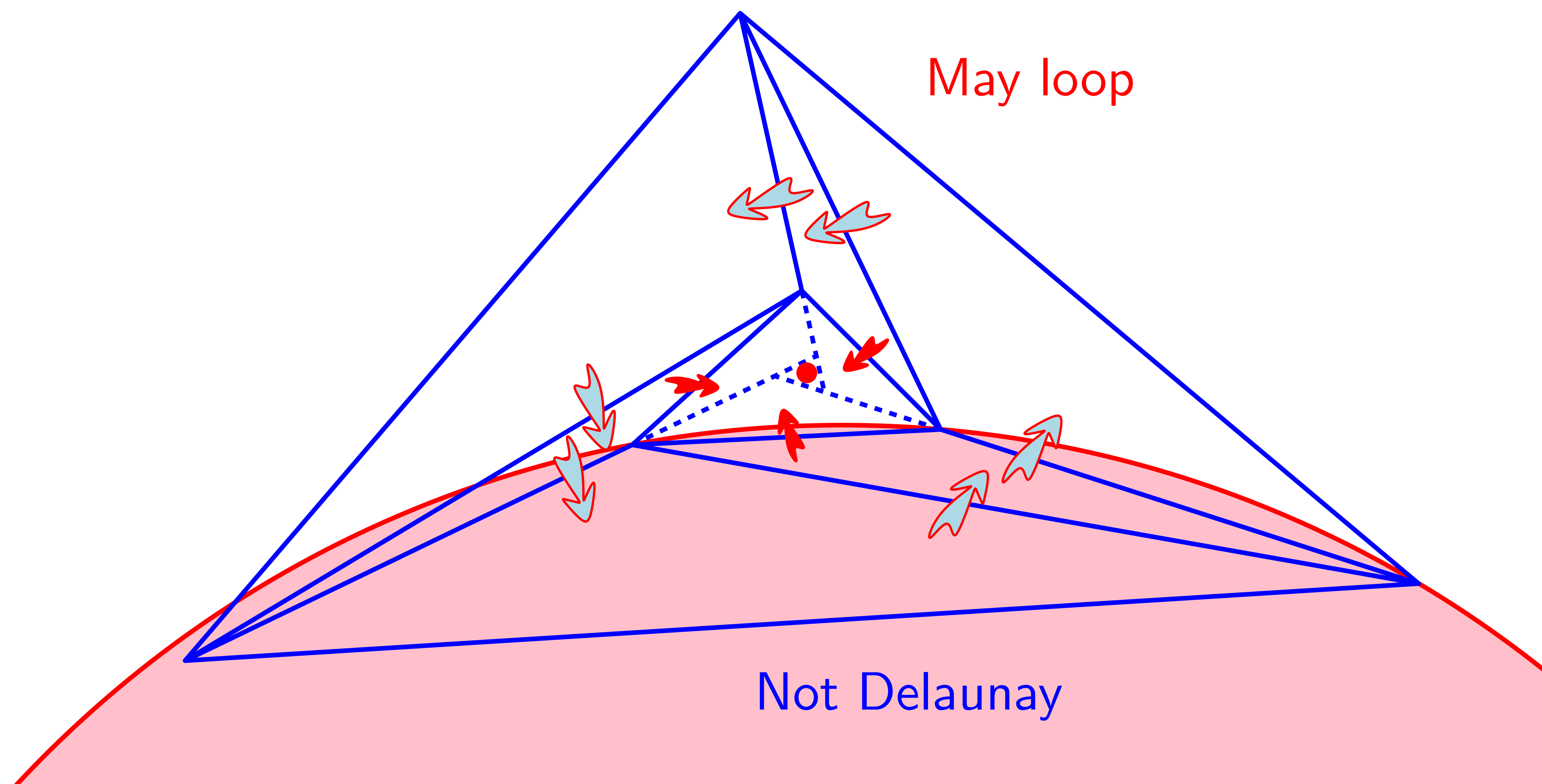
Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?



Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?



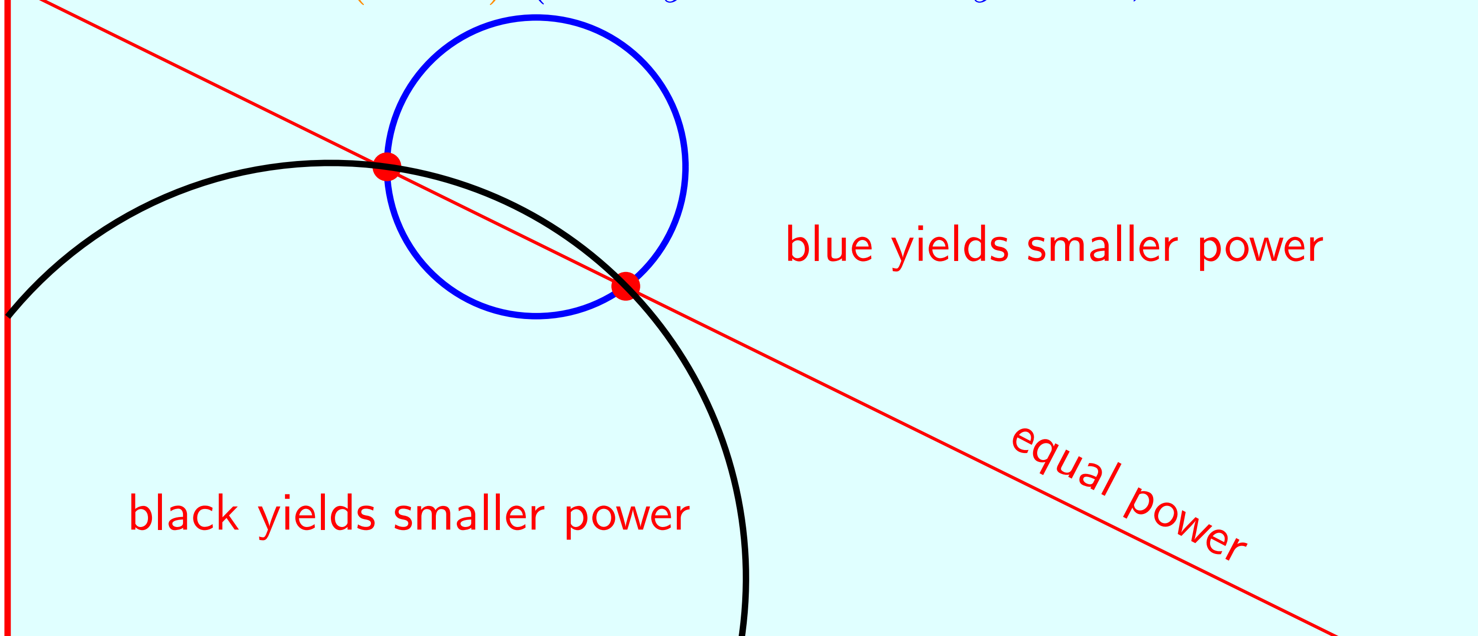
Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?

Delaunay Triangulation: pencils of circles

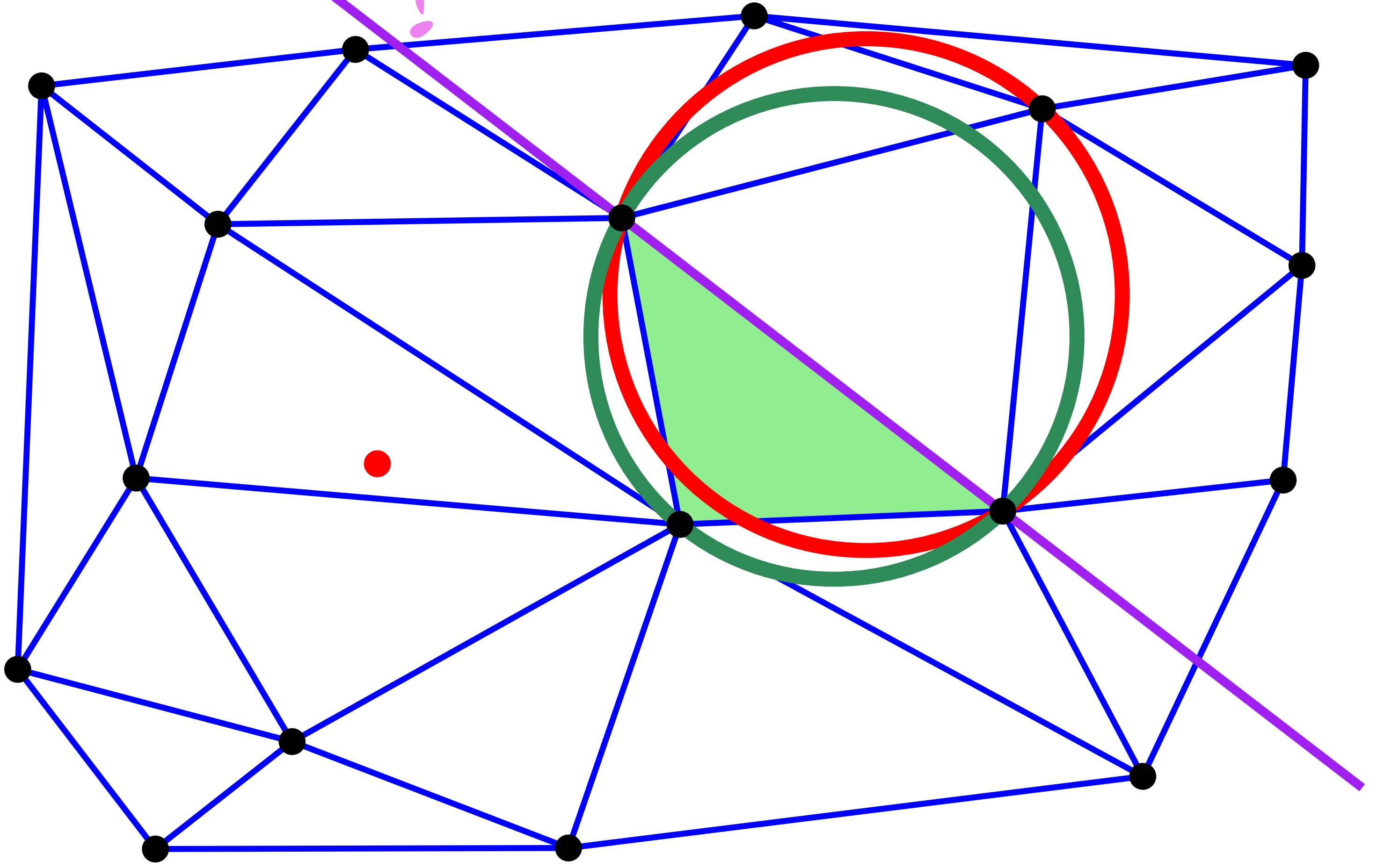
Power of a point w.r.t a circle

$$\lambda \quad (x^2+y^2-2a'x-2b'y+c')$$
$$+(1-\lambda)\ (x^2+y^2-2ax-2by+c\quad)=0$$



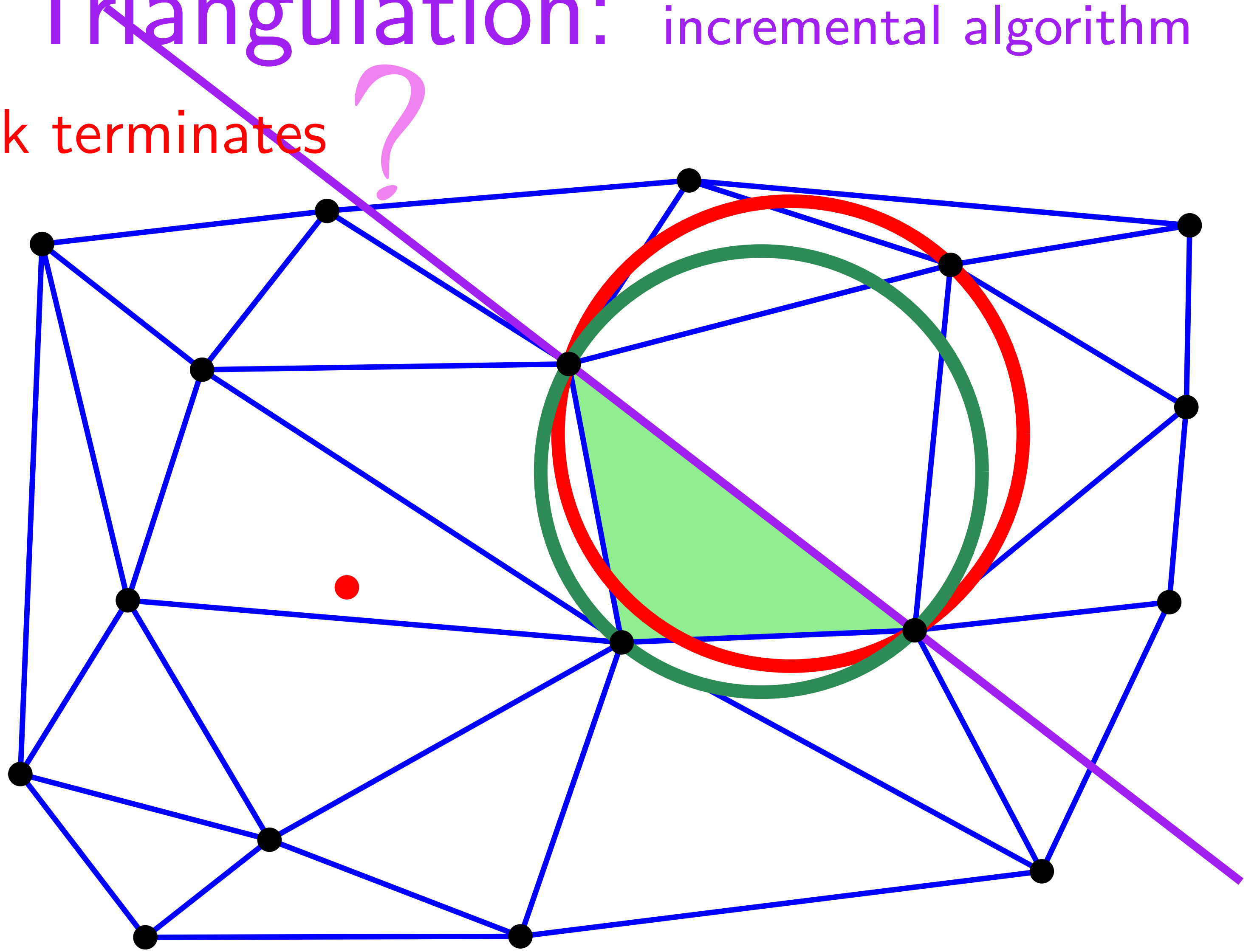
Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?



Delaunay Triangulation: incremental algorithm

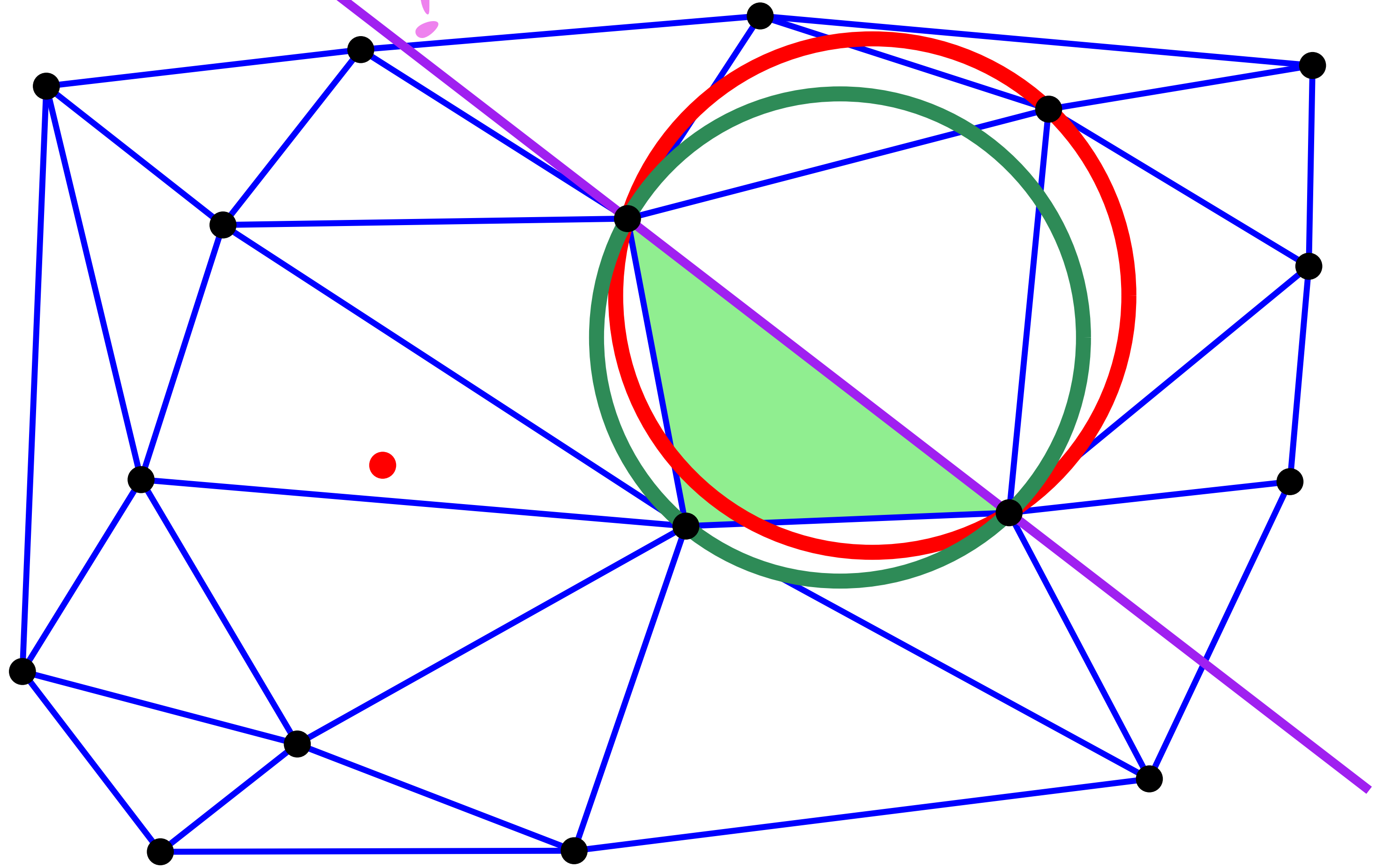
Visibility walk terminates ?



Green power $<$ Red power

Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?

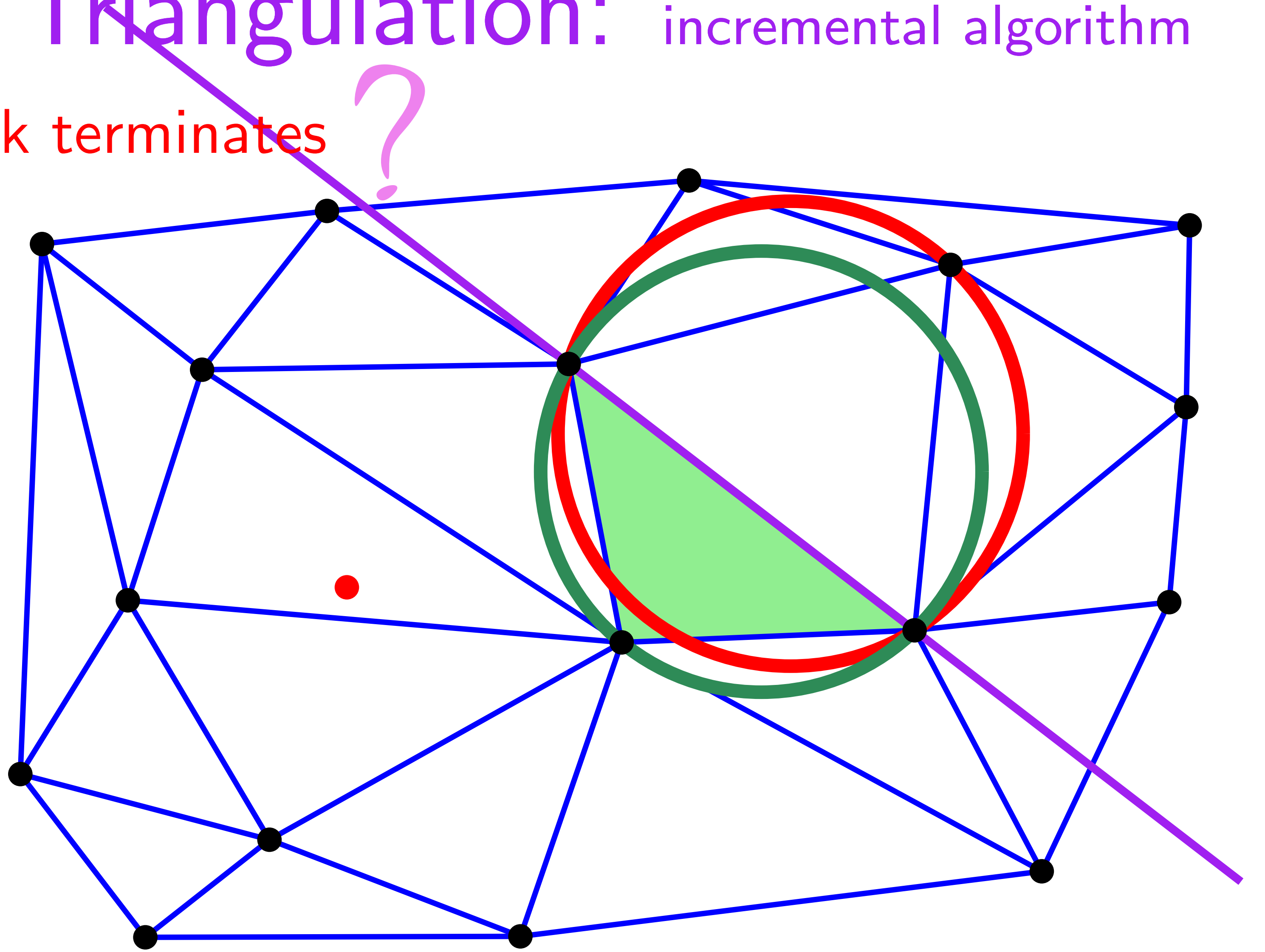


Green power < Red power

Power decreases

Delaunay Triangulation: incremental algorithm

Visibility walk terminates ?



Green power $<$ Red power

Power decreases

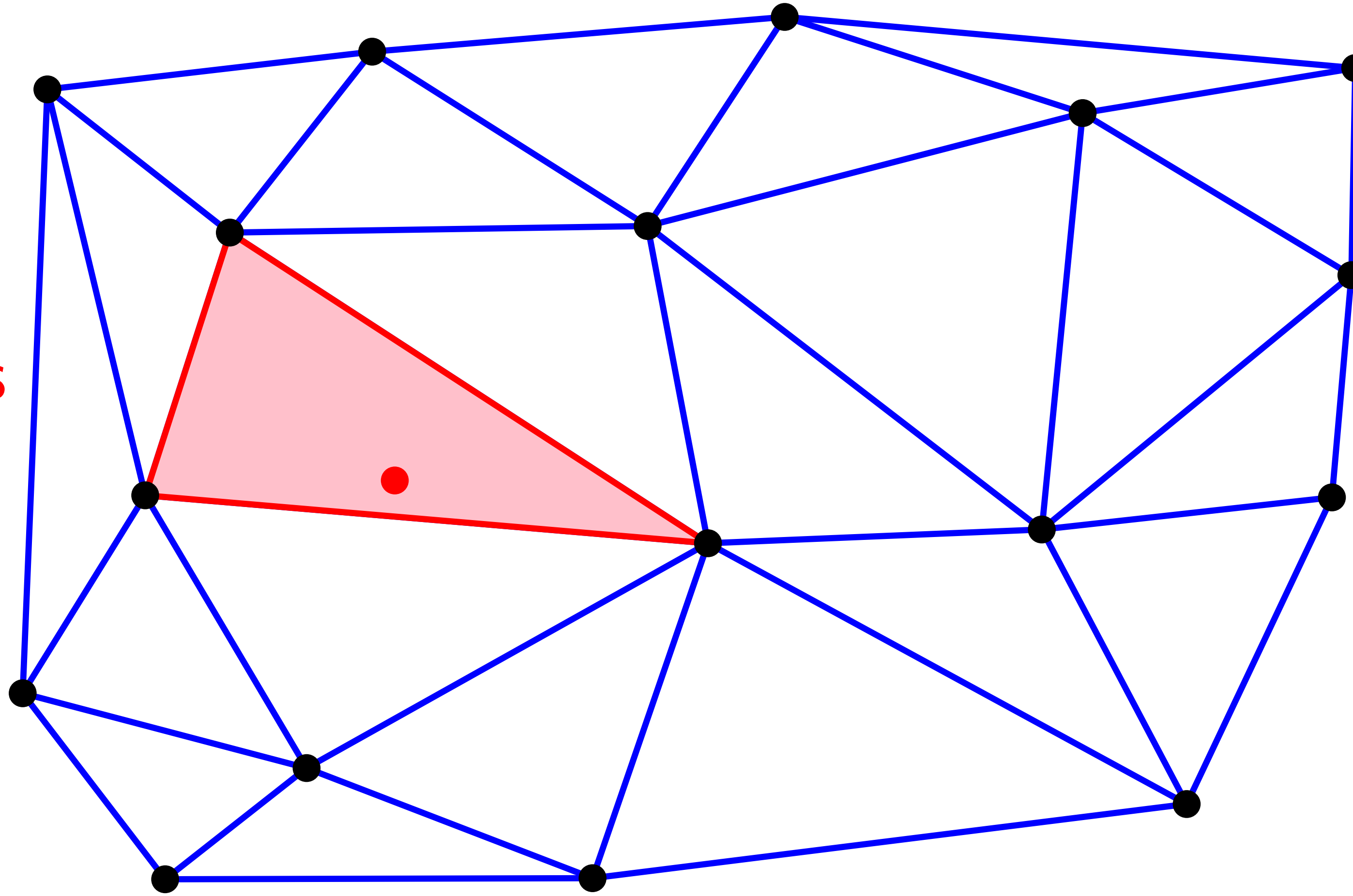
Visibility walk terminates

Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

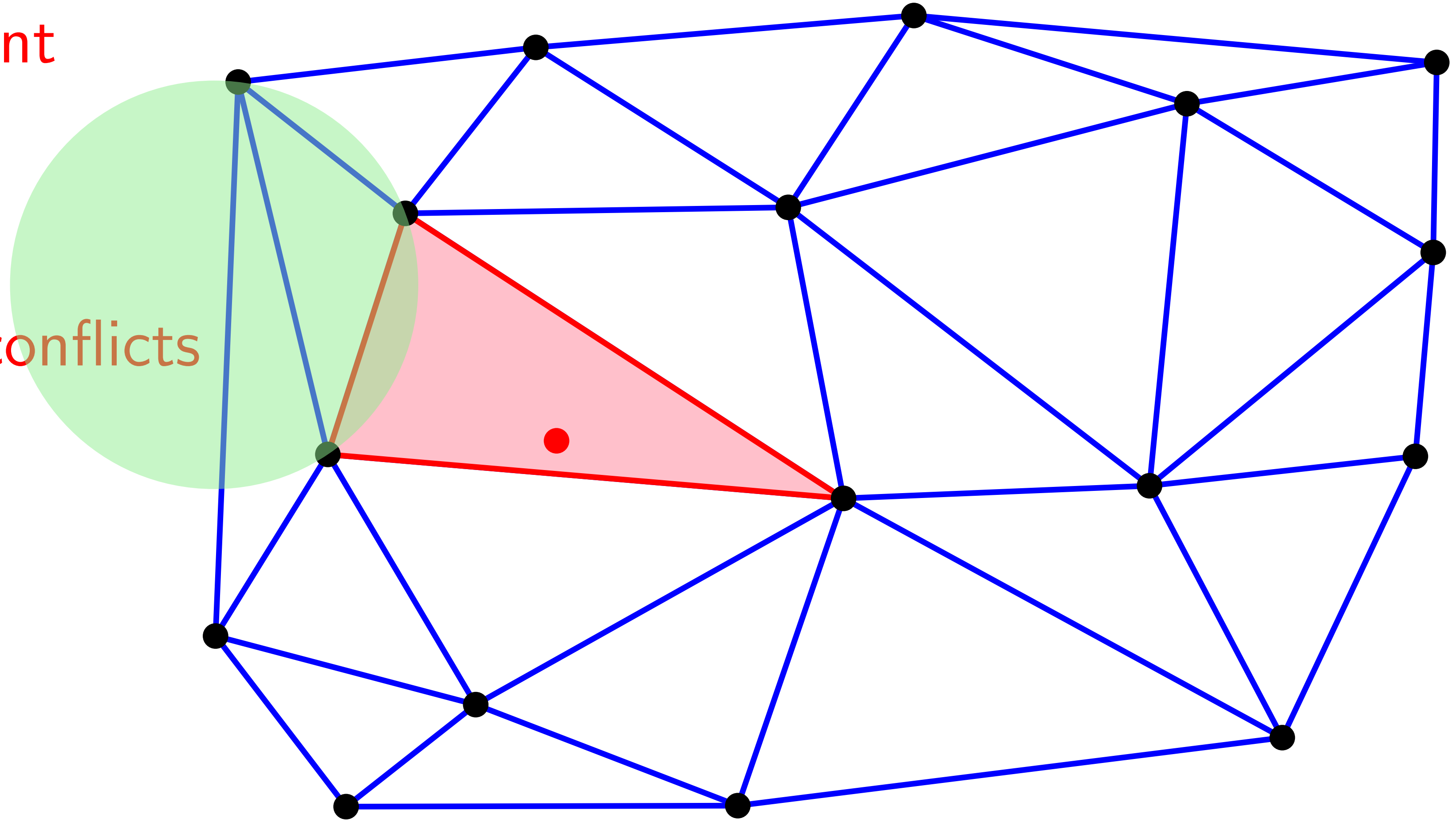


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

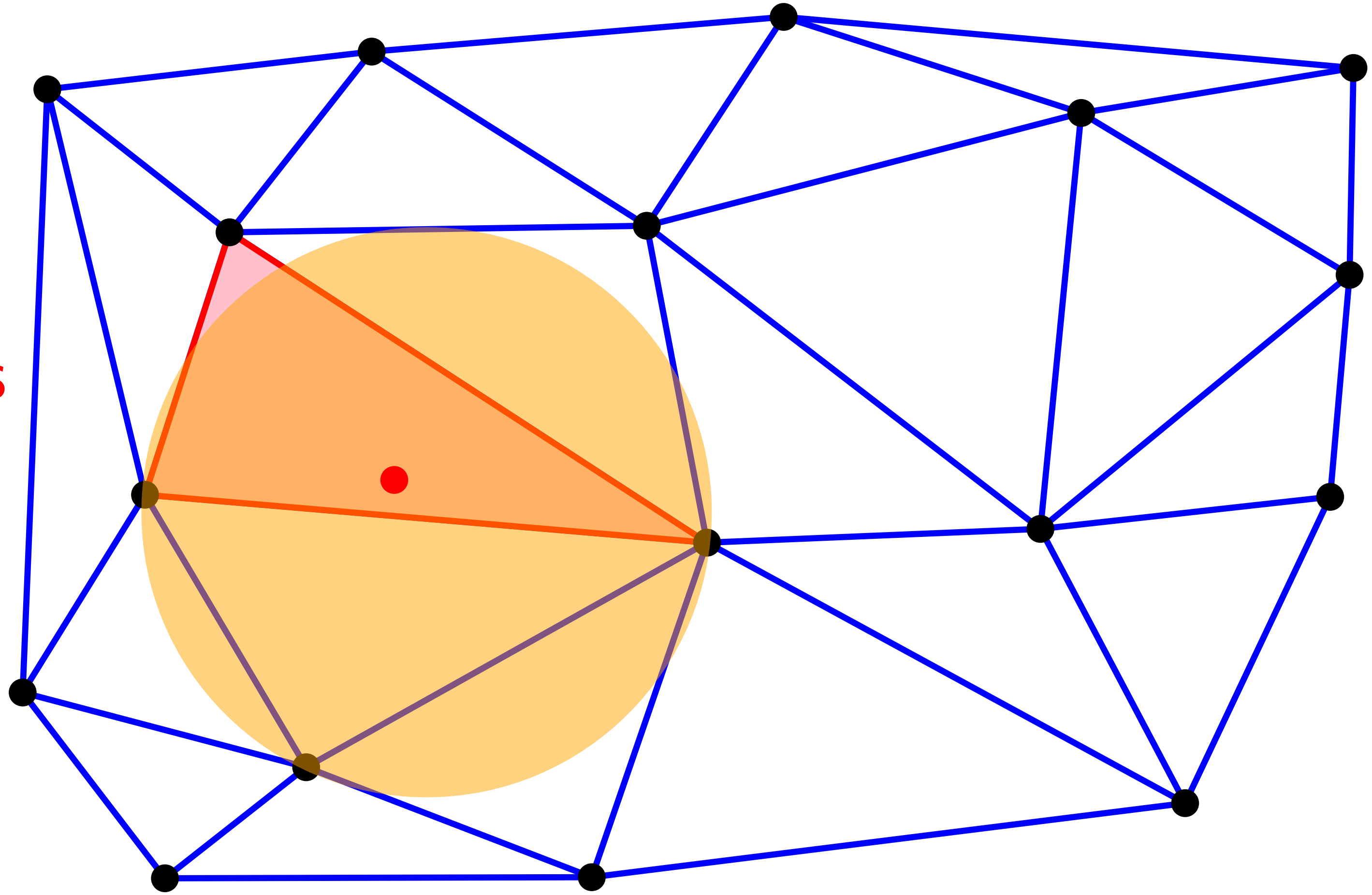


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

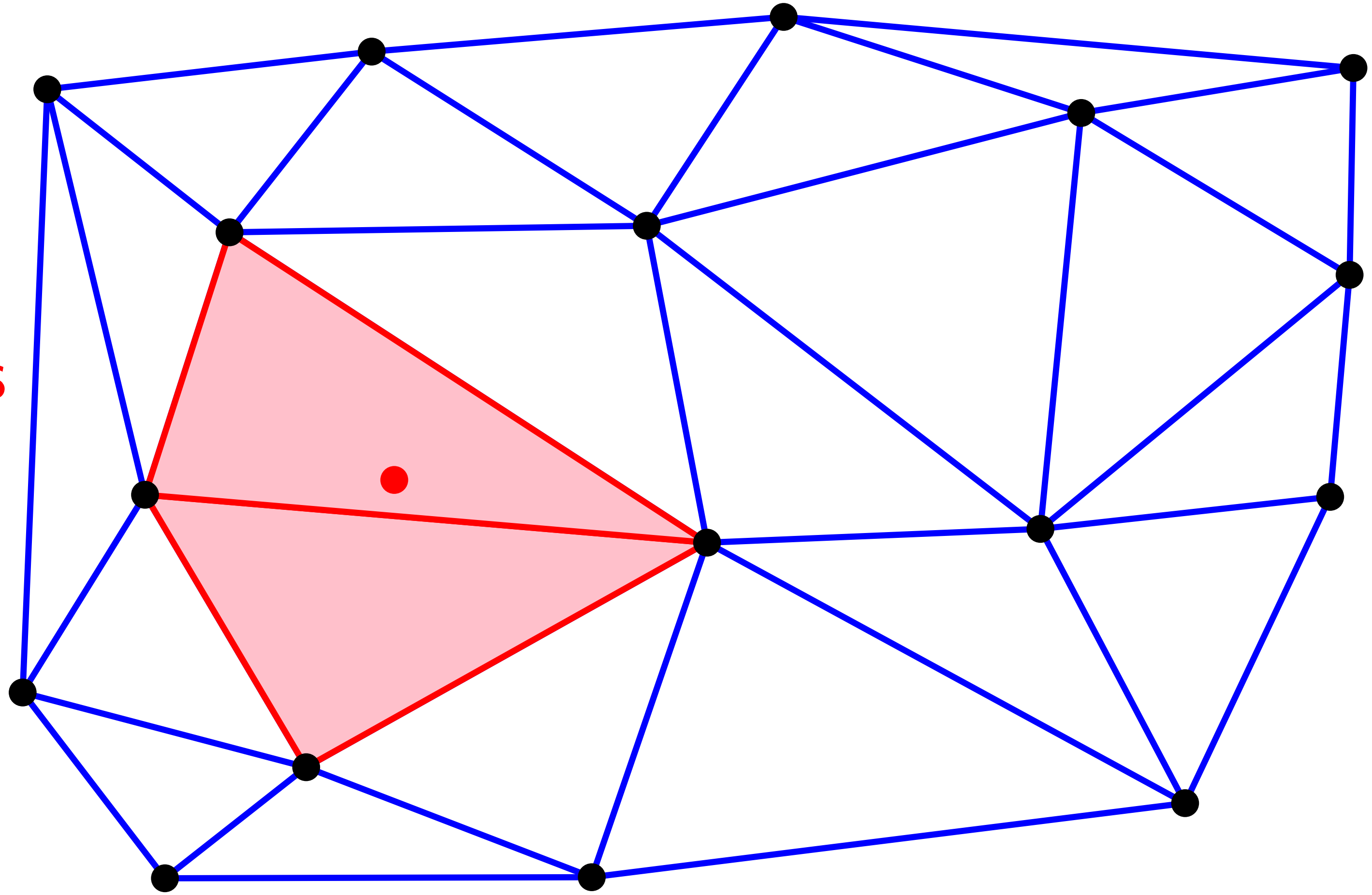


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

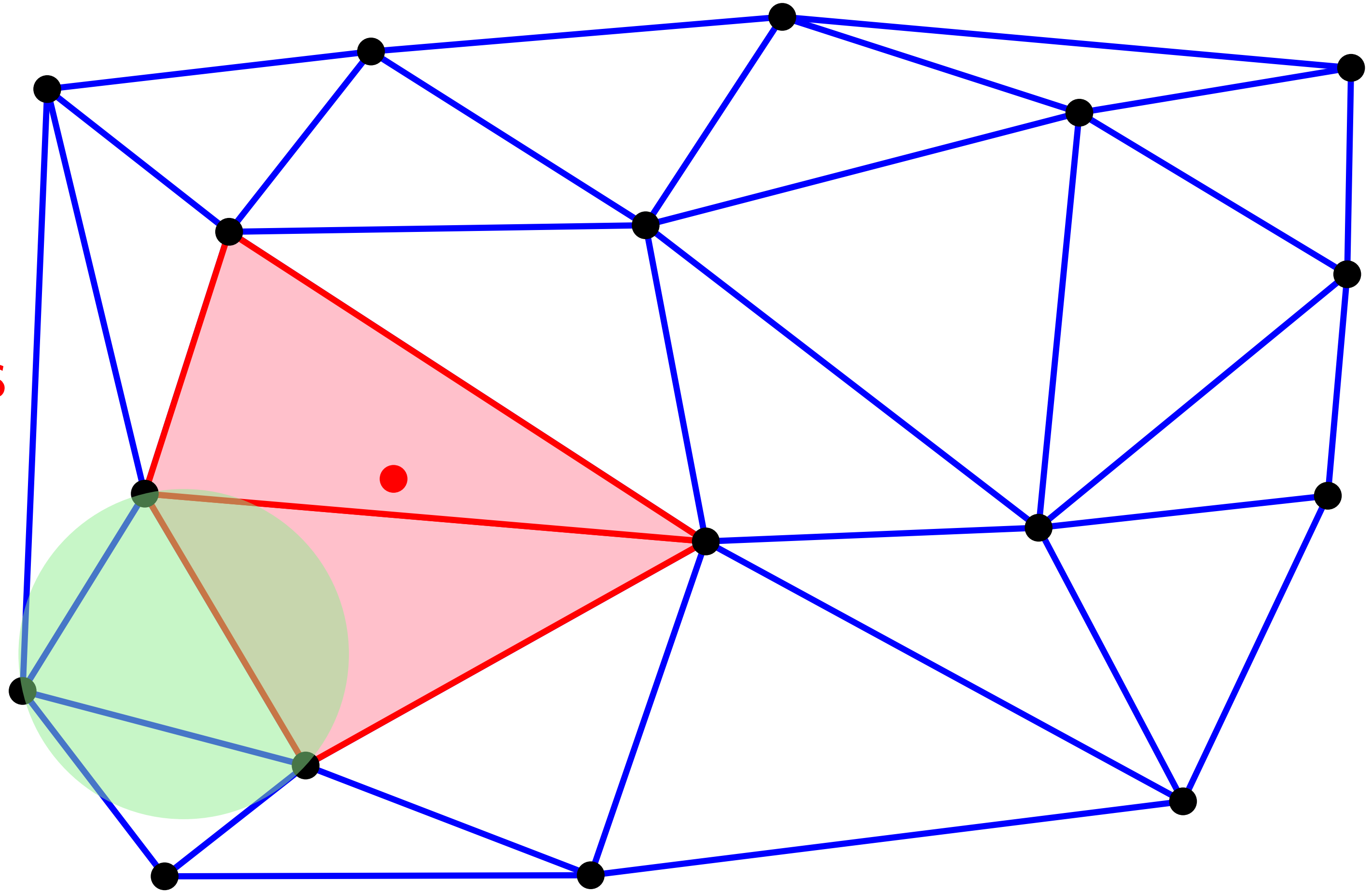


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

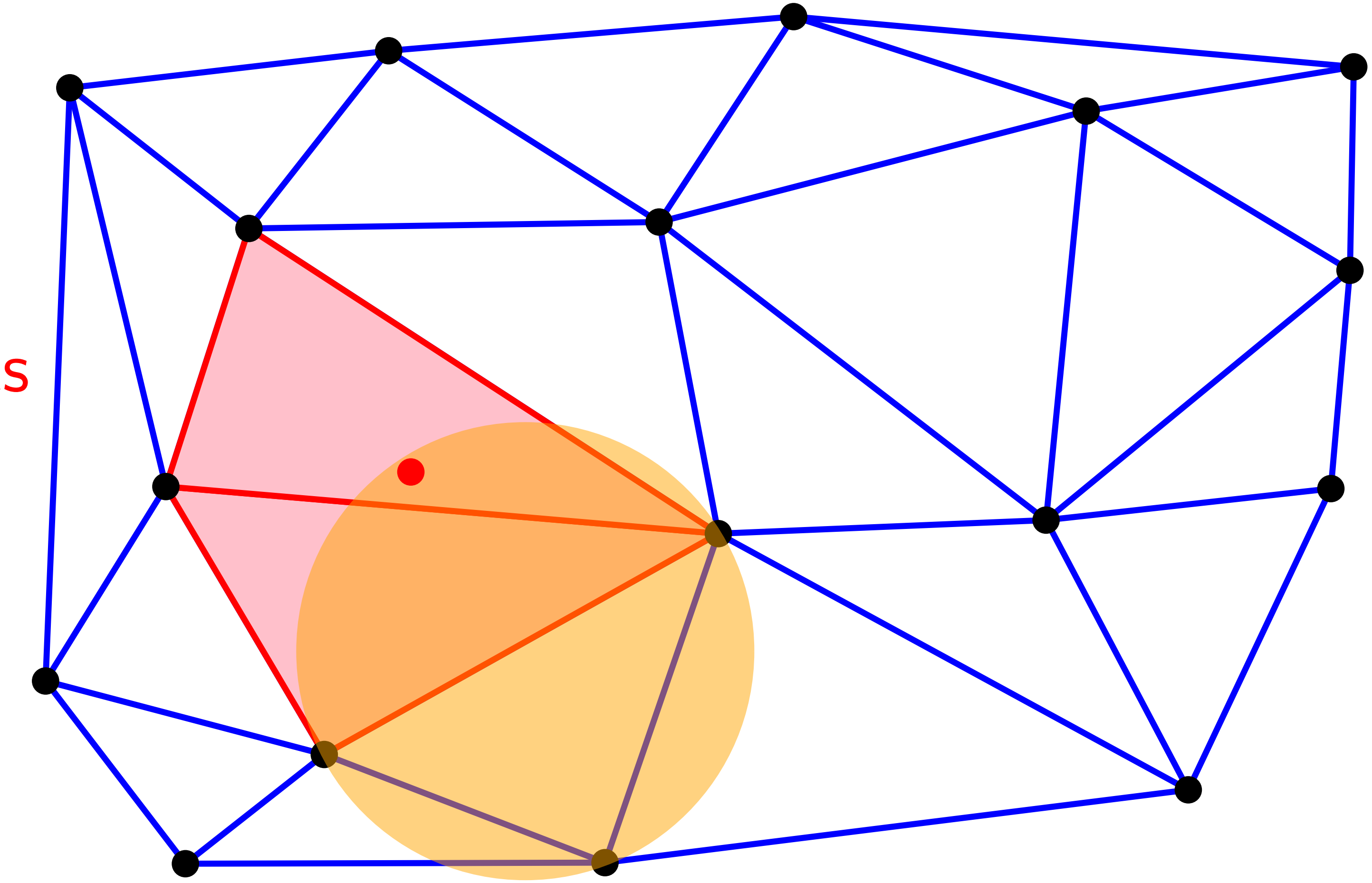


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

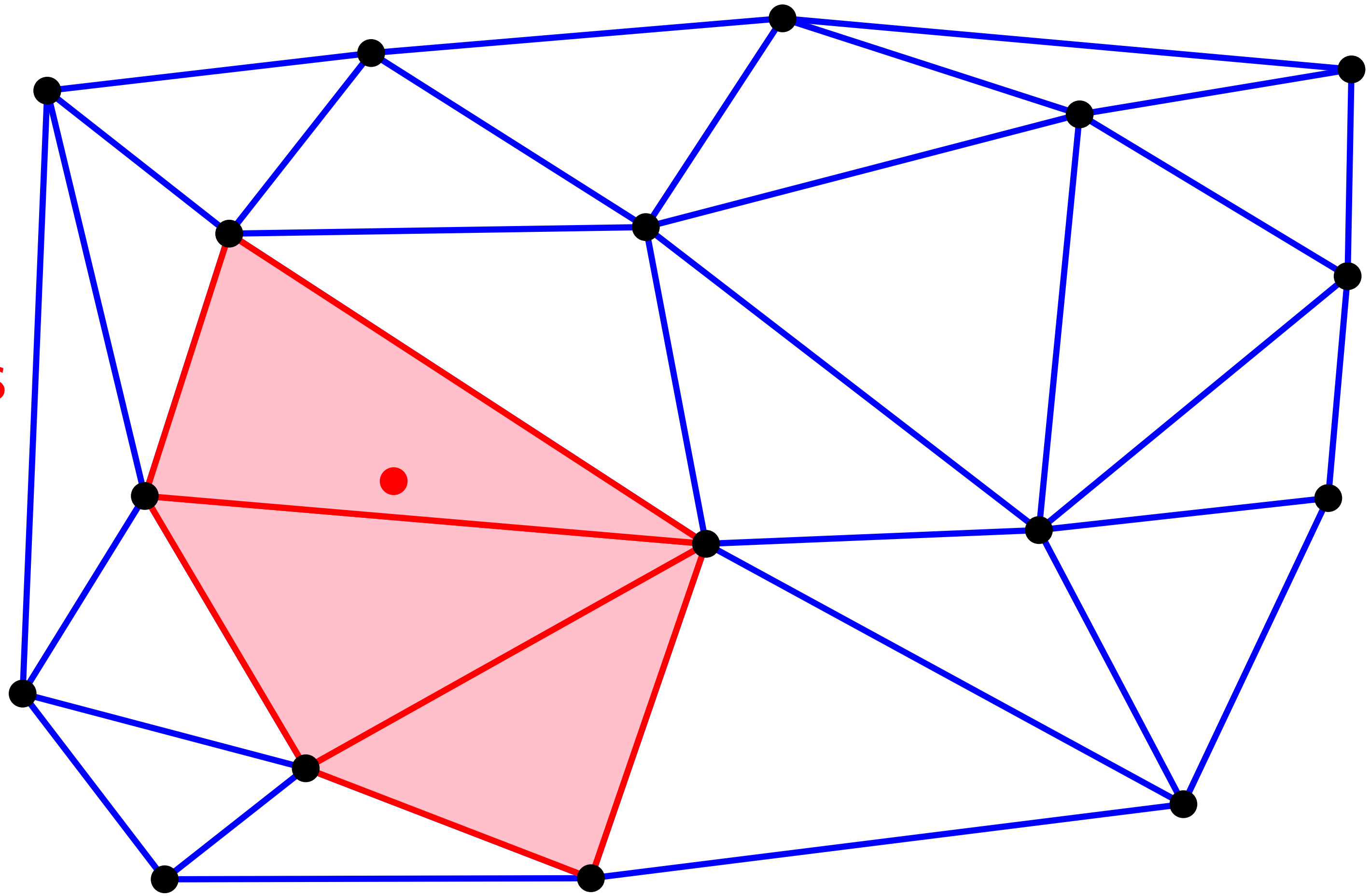


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

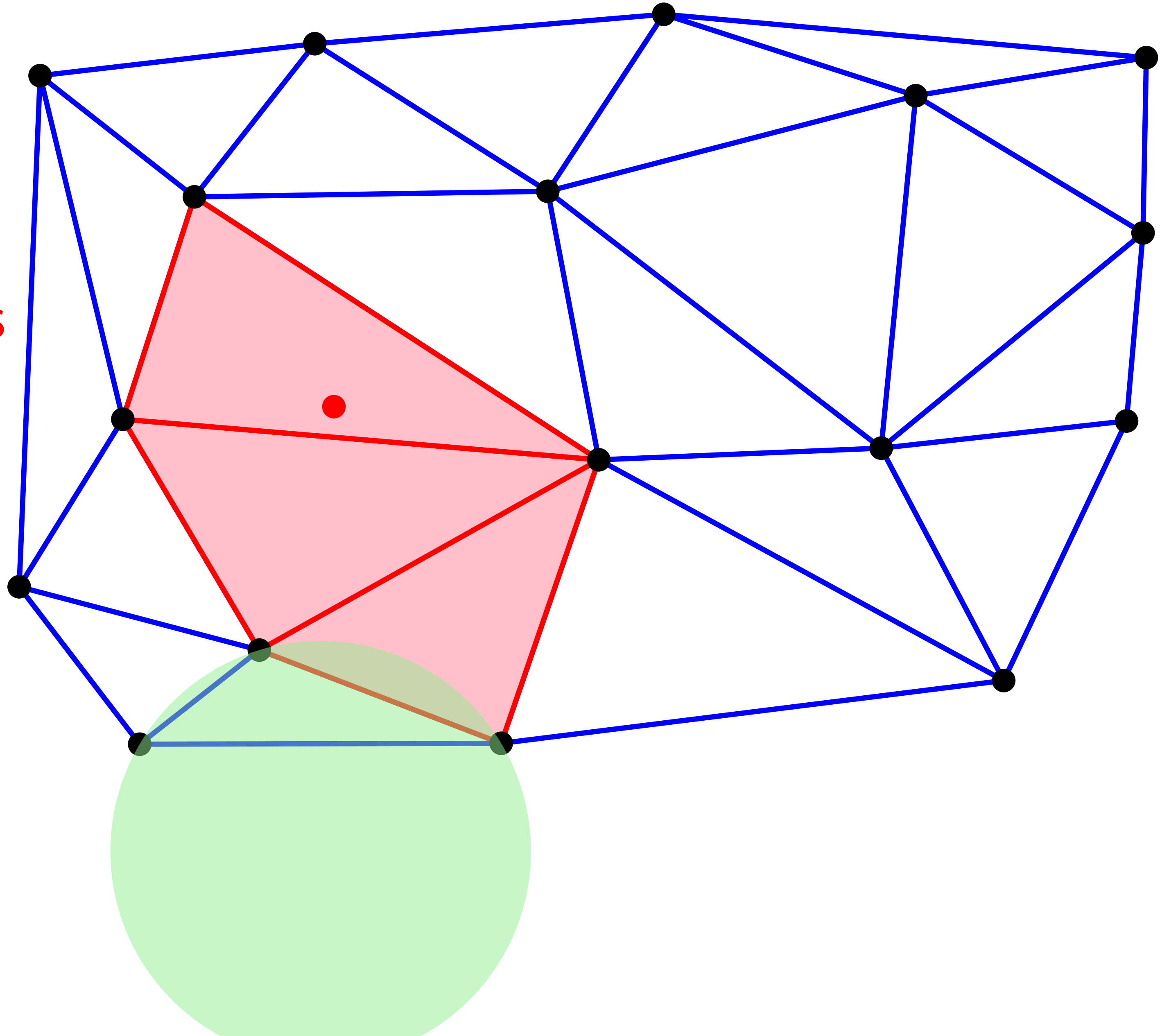


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

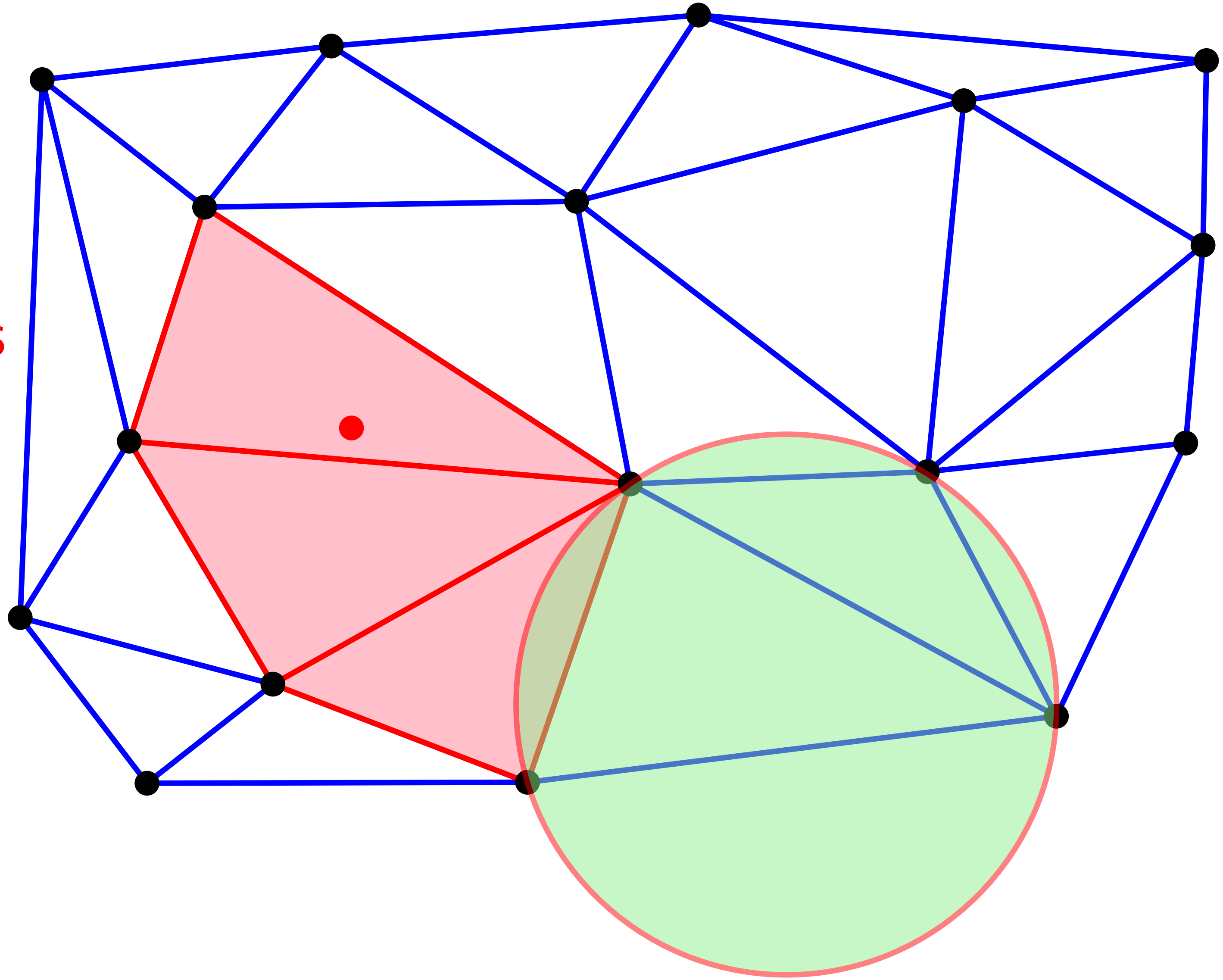


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

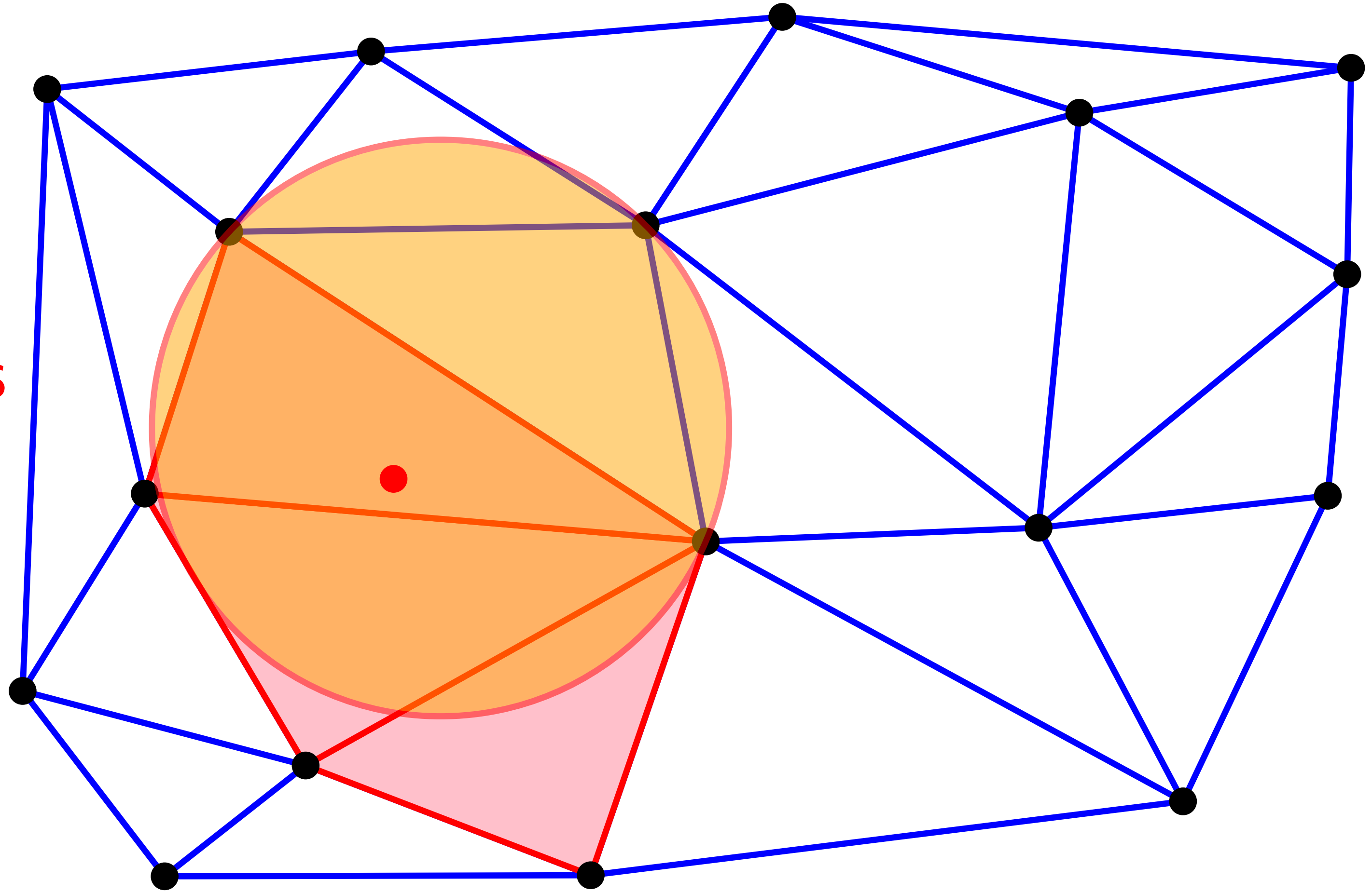


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

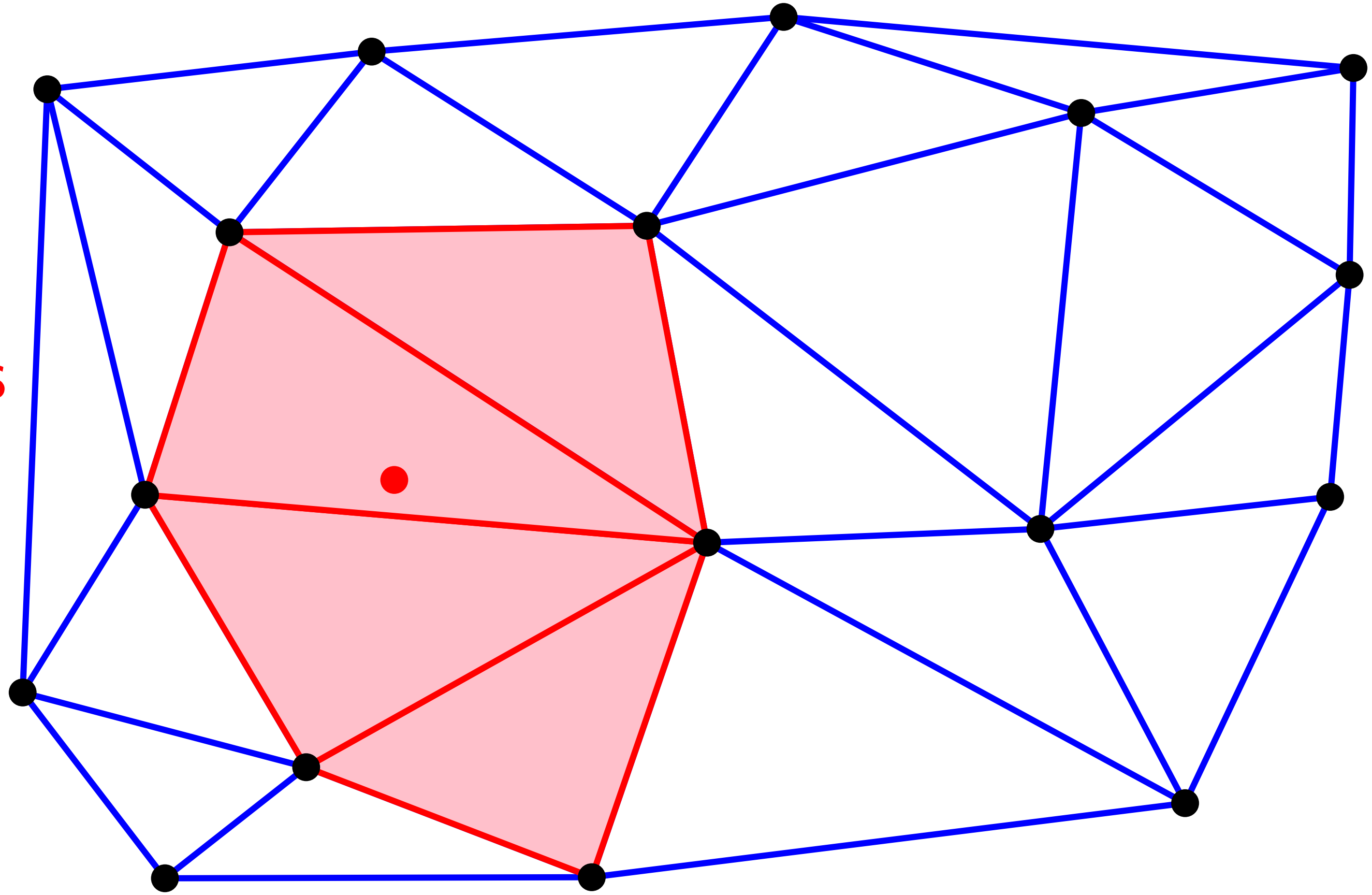


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

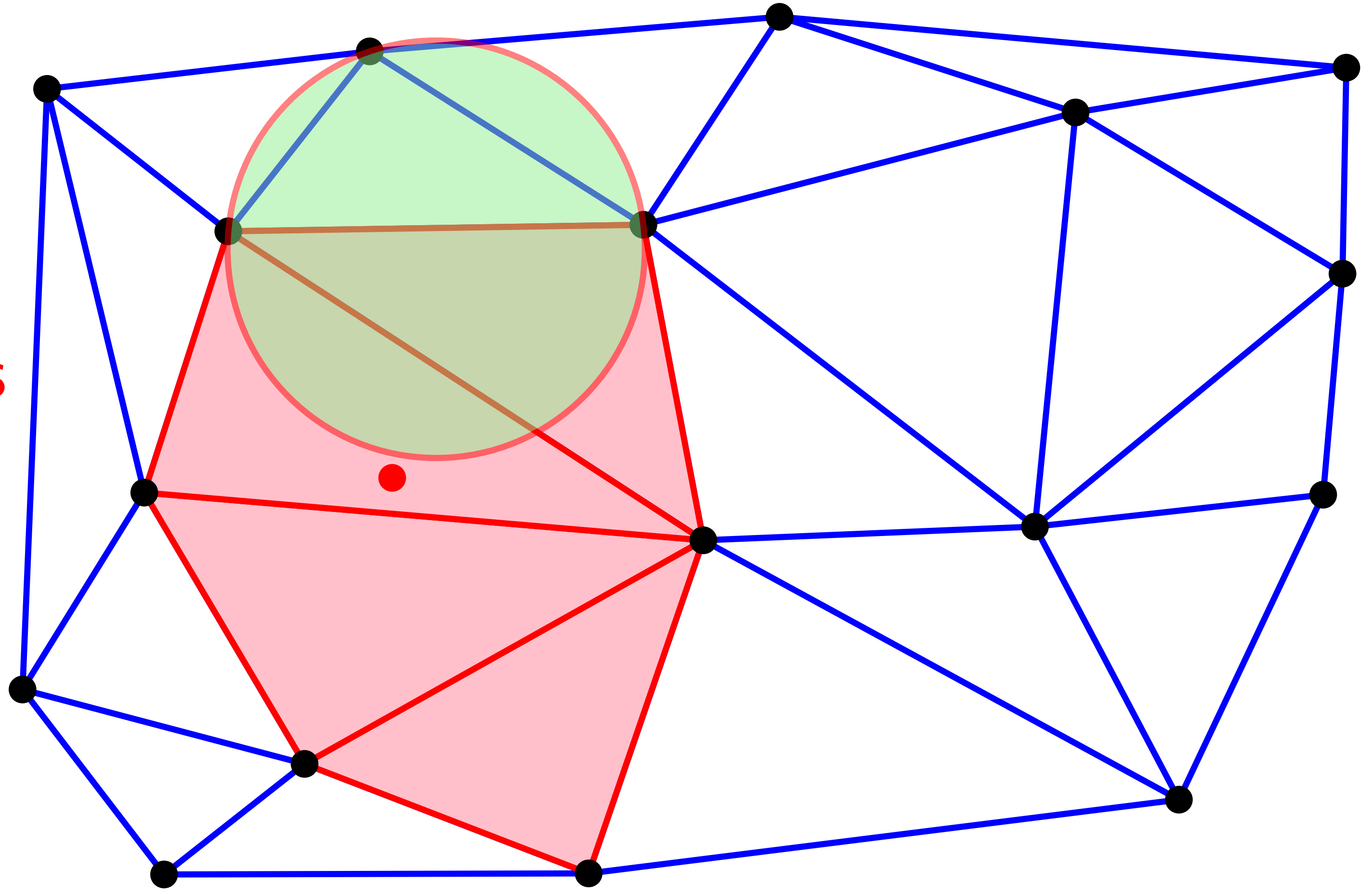


Delaunay Triangulation: incremental algorithm

New point

Locate

Search conflicts

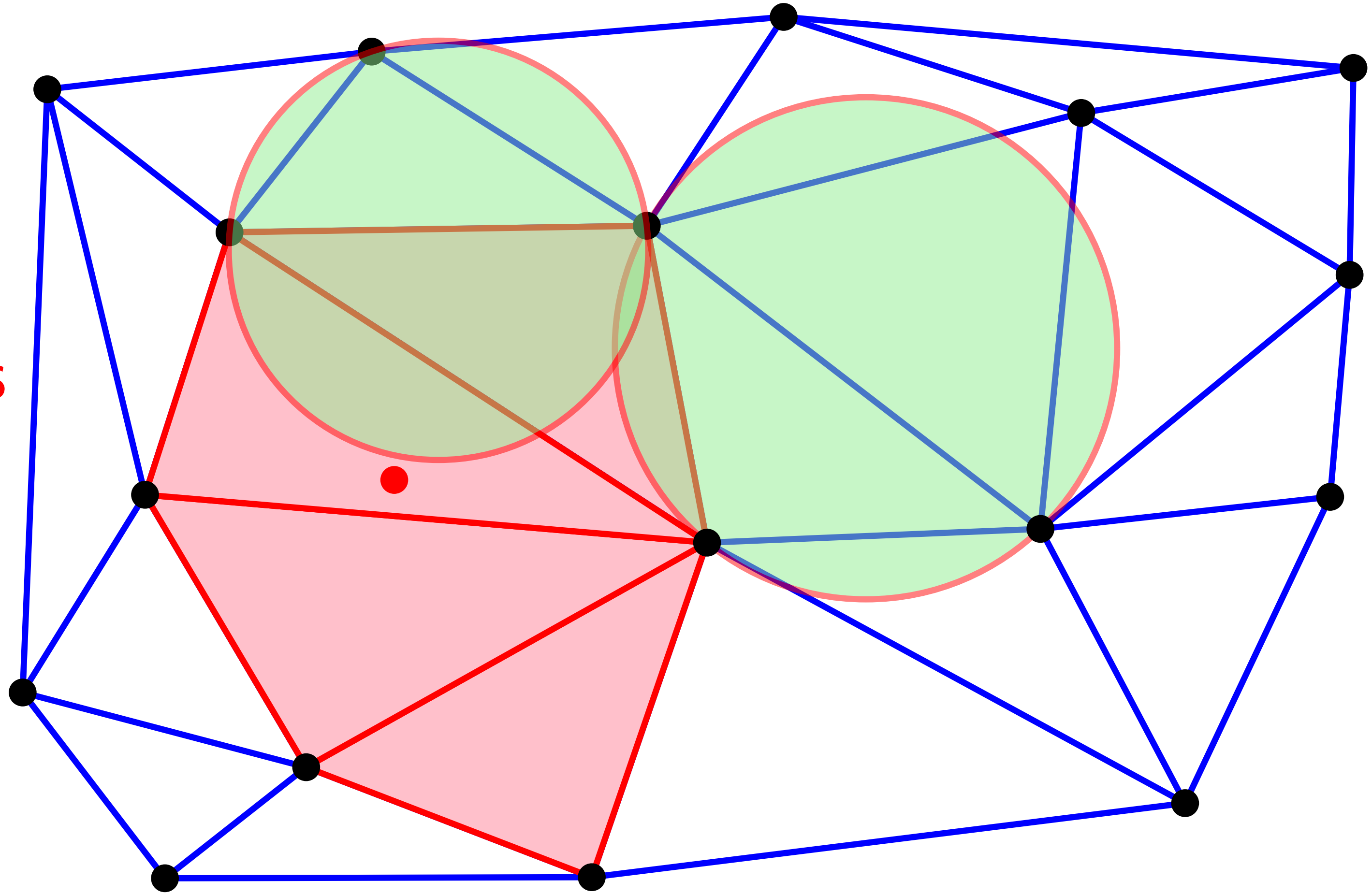


Delaunay Triangulation: incremental algorithm

New point

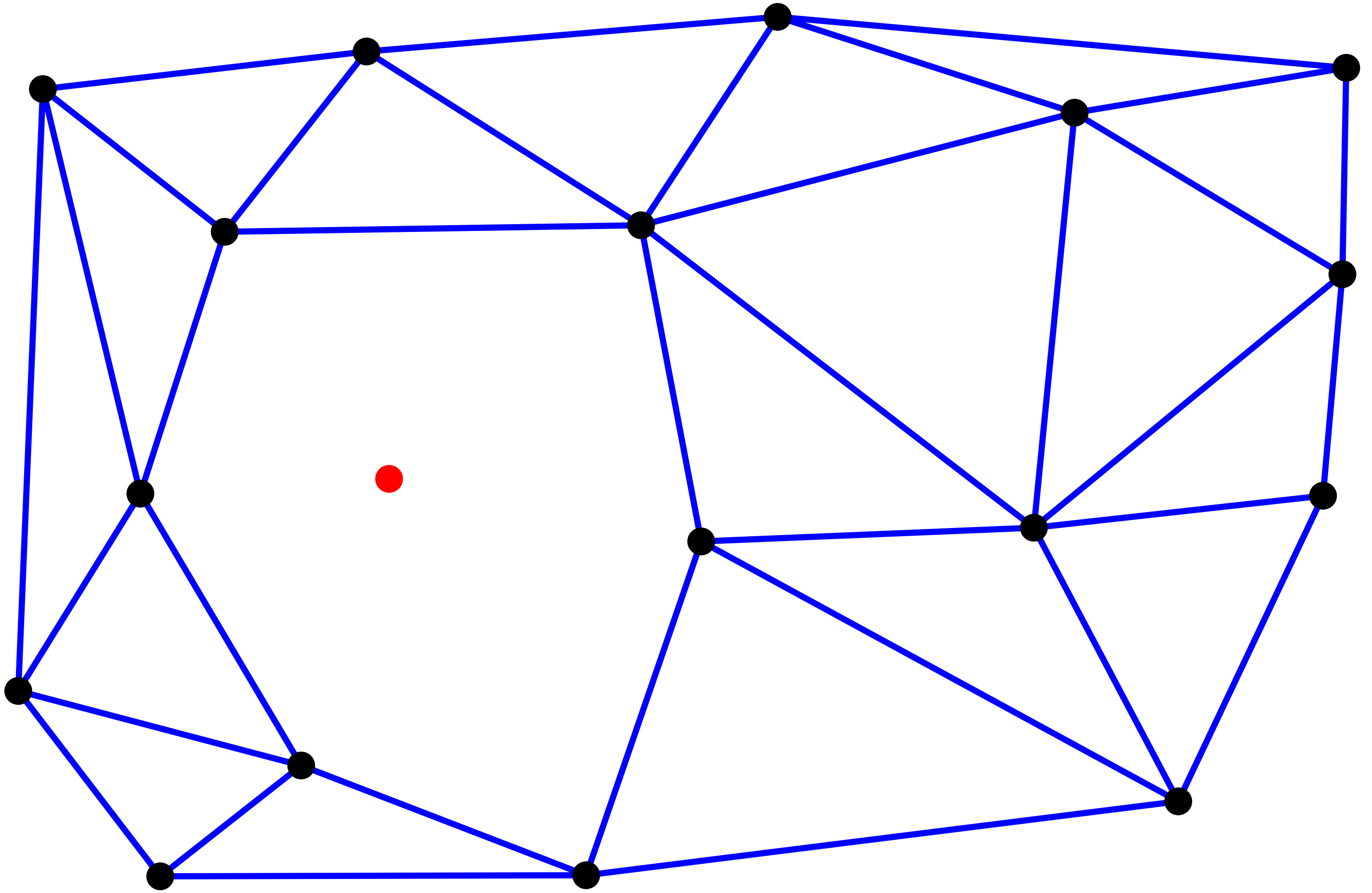
Locate

Search conflicts



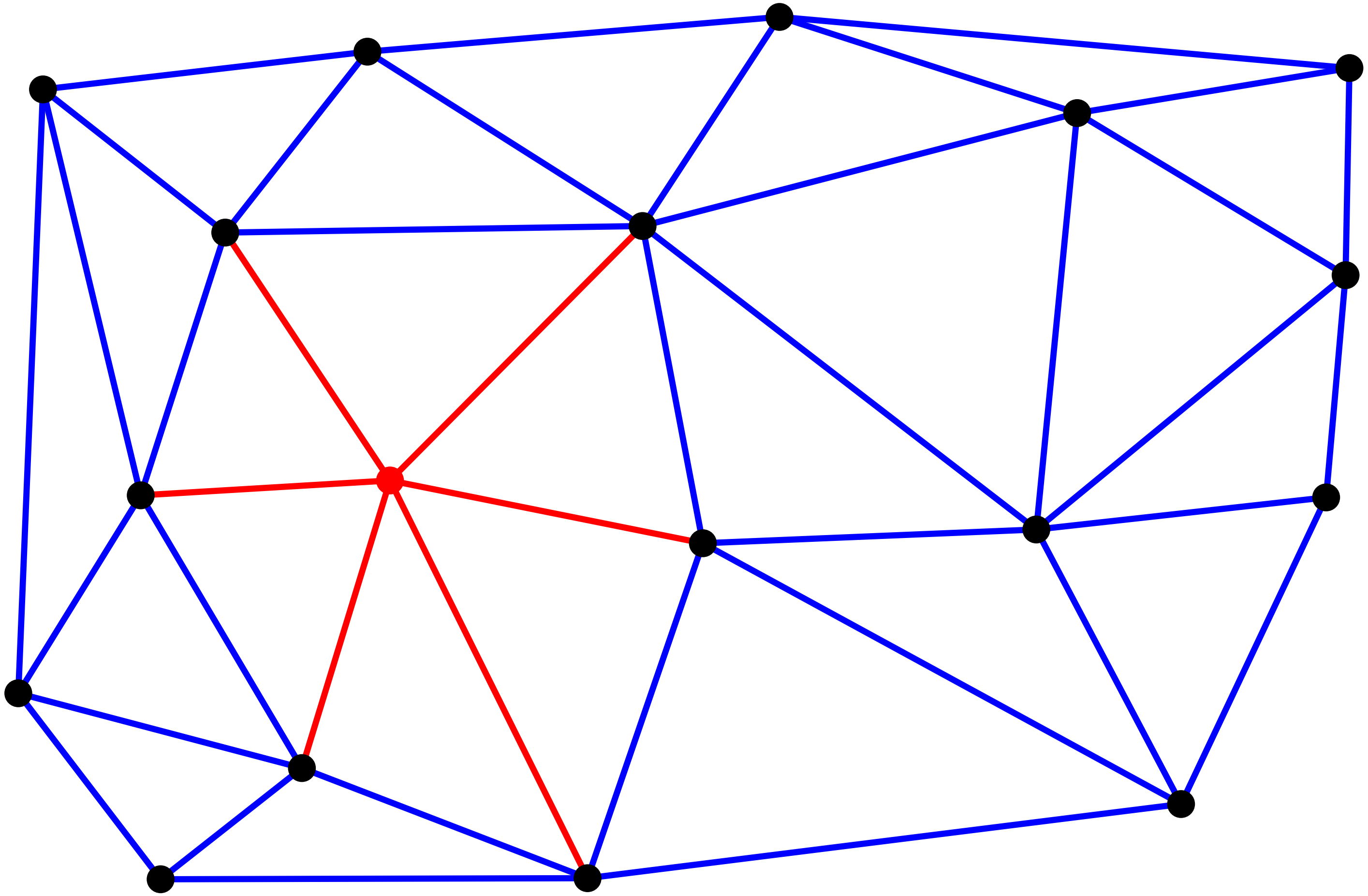
Delaunay Triangulation: incremental algorithm

New point



Delaunay Triangulation: incremental algorithm

New point



Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts

Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts $\sharp$ triangles in conflict

$\sharp$ triangles neighboring triangles in conflict

Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts $\sharp$ triangles in conflict

$\sharp$ triangles neighboring triangles in conflict

degree of new point in new triangulation

$< n$

Delaunay Triangulation: incremental algorithm

Complexity

Locate

Walk may visit all triangles

$< 2n$

Search conflicts

degree of new point in new triangulation

$< n$

Delaunay Triangulation: incremental algorithm

Complexity

Locate $O(n)$ per insertion

Search conflicts

Delaunay Triangulation: incremental algorithm

Complexity

Locate $O(n)$ per insertion

Search conflicts

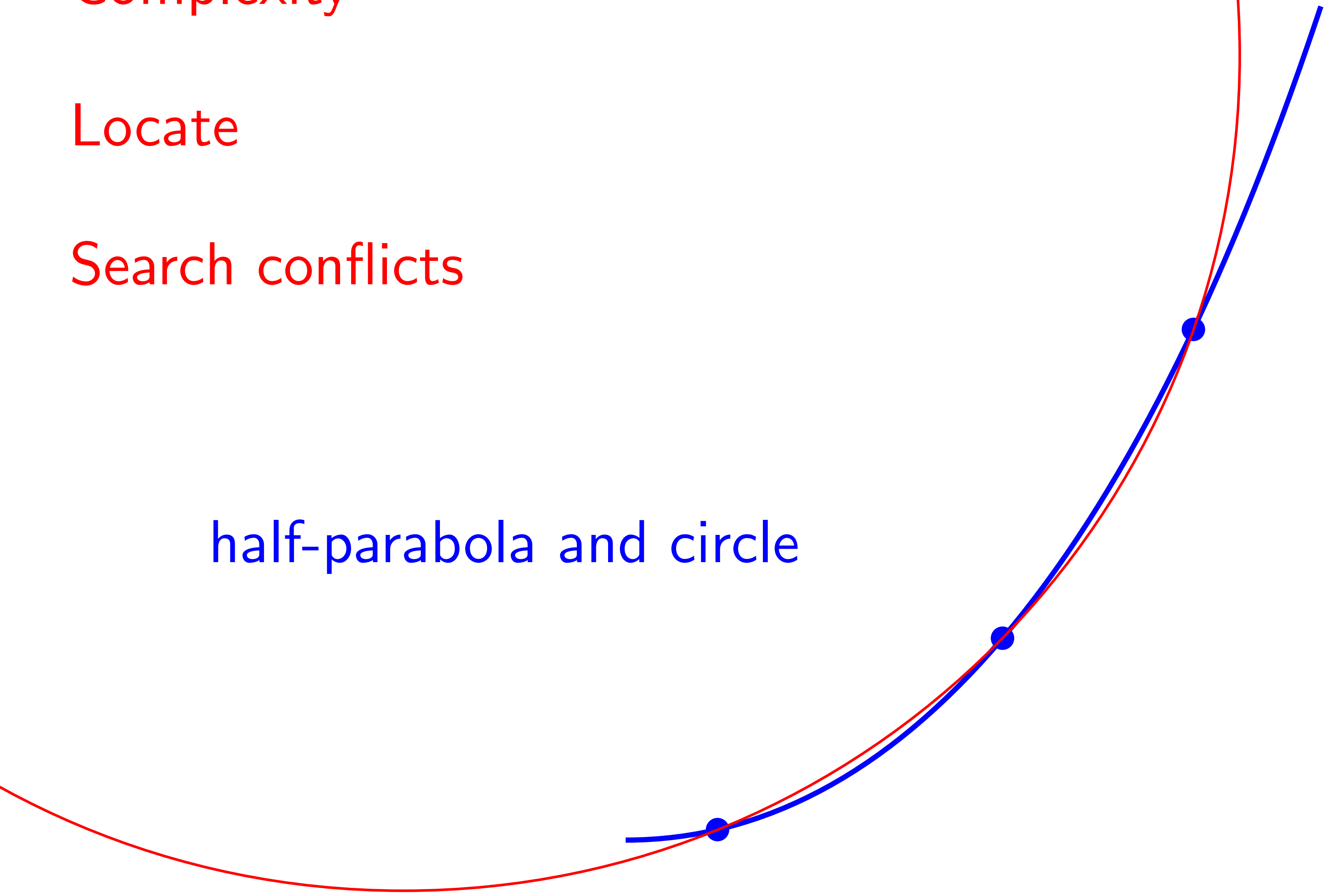
$O(n^2)$ for the whole construction

Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts

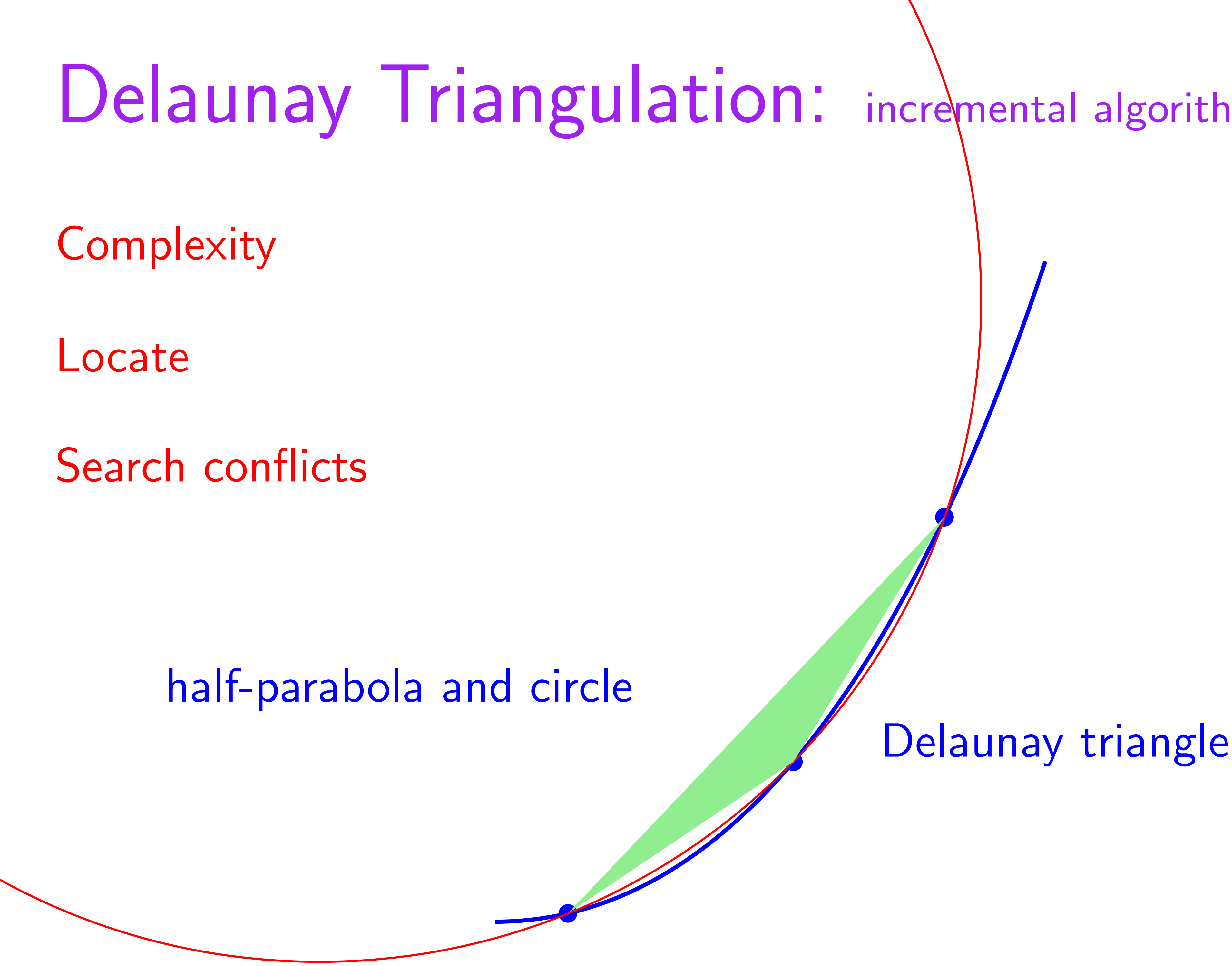


Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts

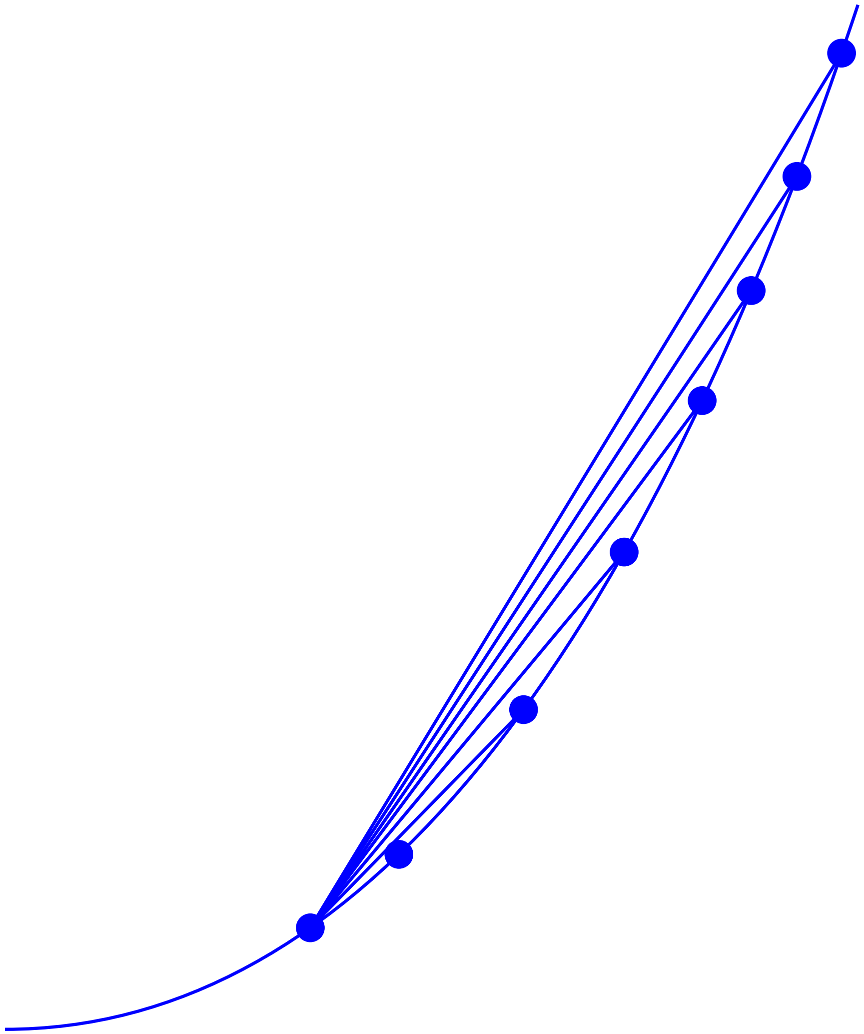


Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts

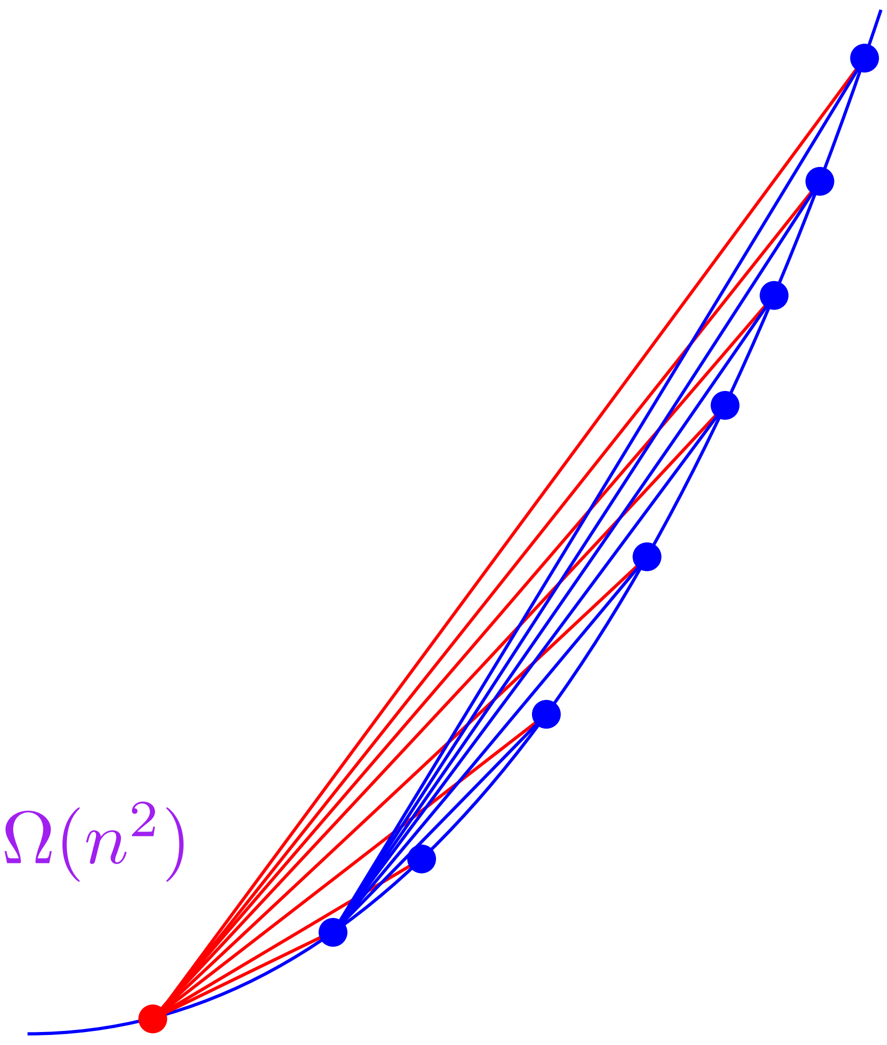


Delaunay Triangulation: incremental algorithm

Complexity

Locate

Search conflicts



Insertion: $\Omega(n)$

Whole construction: $\Omega(n^2)$

Delaunay Triangulation: incremental algorithm

In practice

Complexity

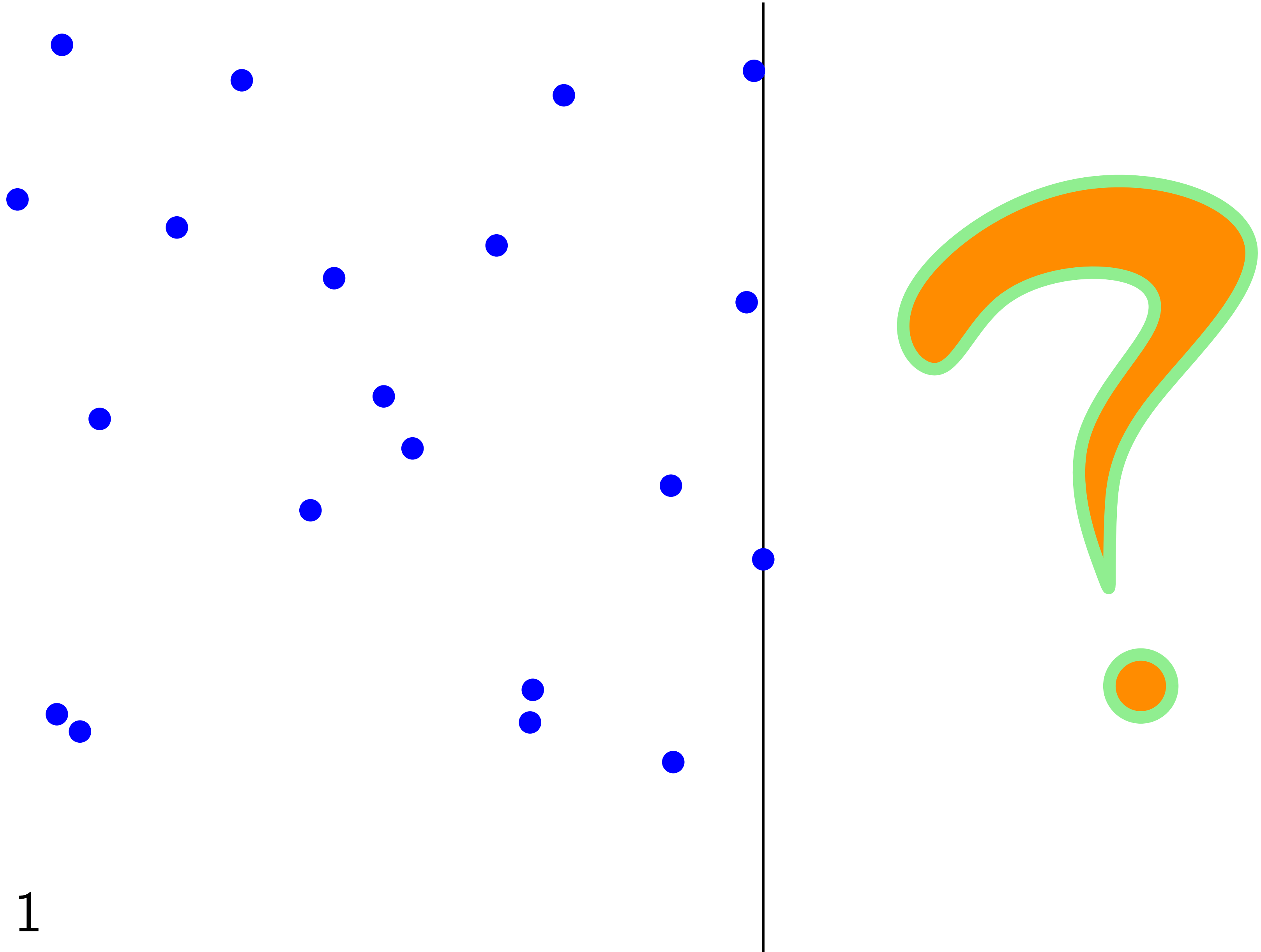
Locate — Many possibilities (walk, Delaunay hierarchy)

Search conflicts — Randomized

Teaser randomization lecture

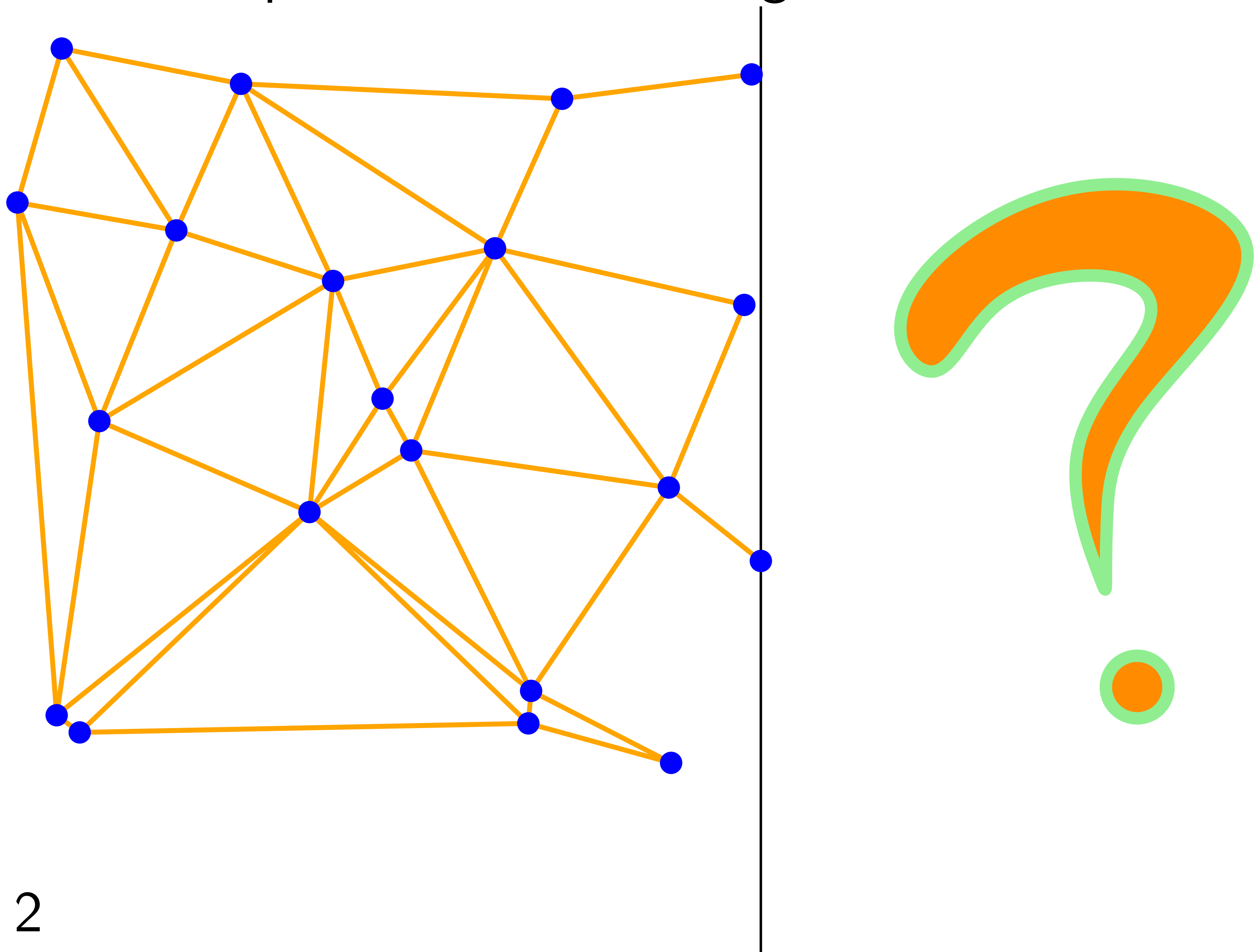
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



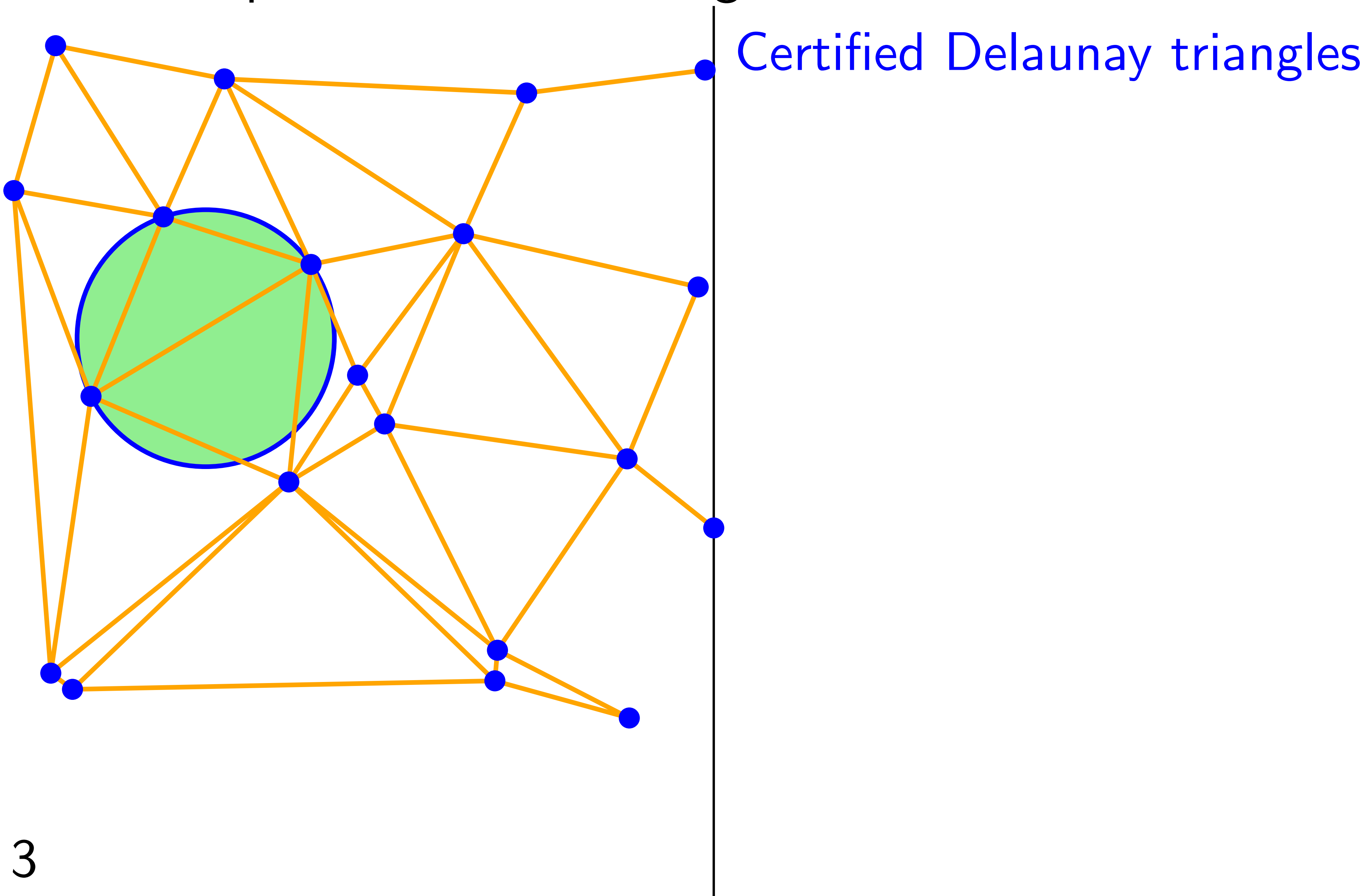
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



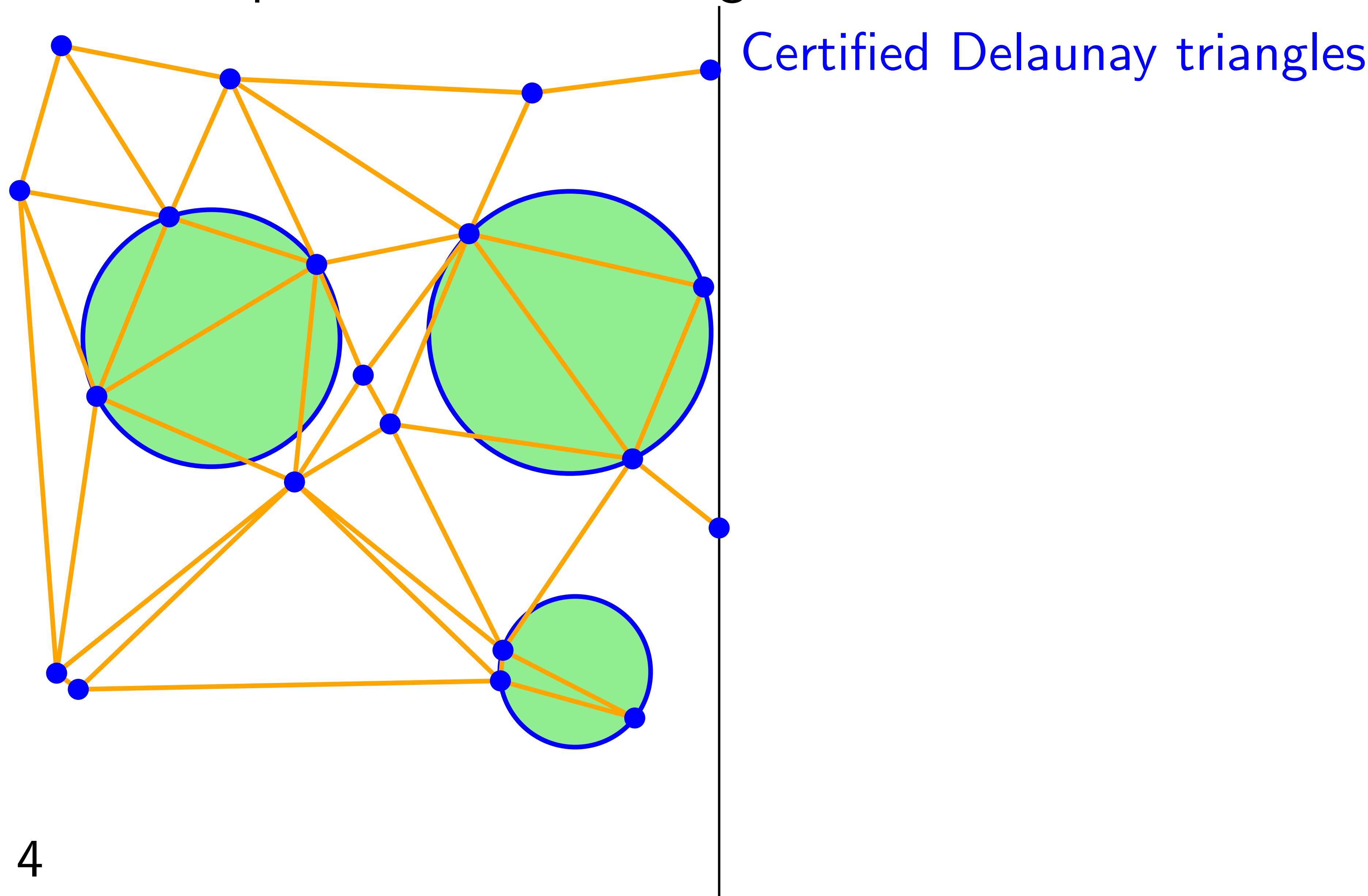
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



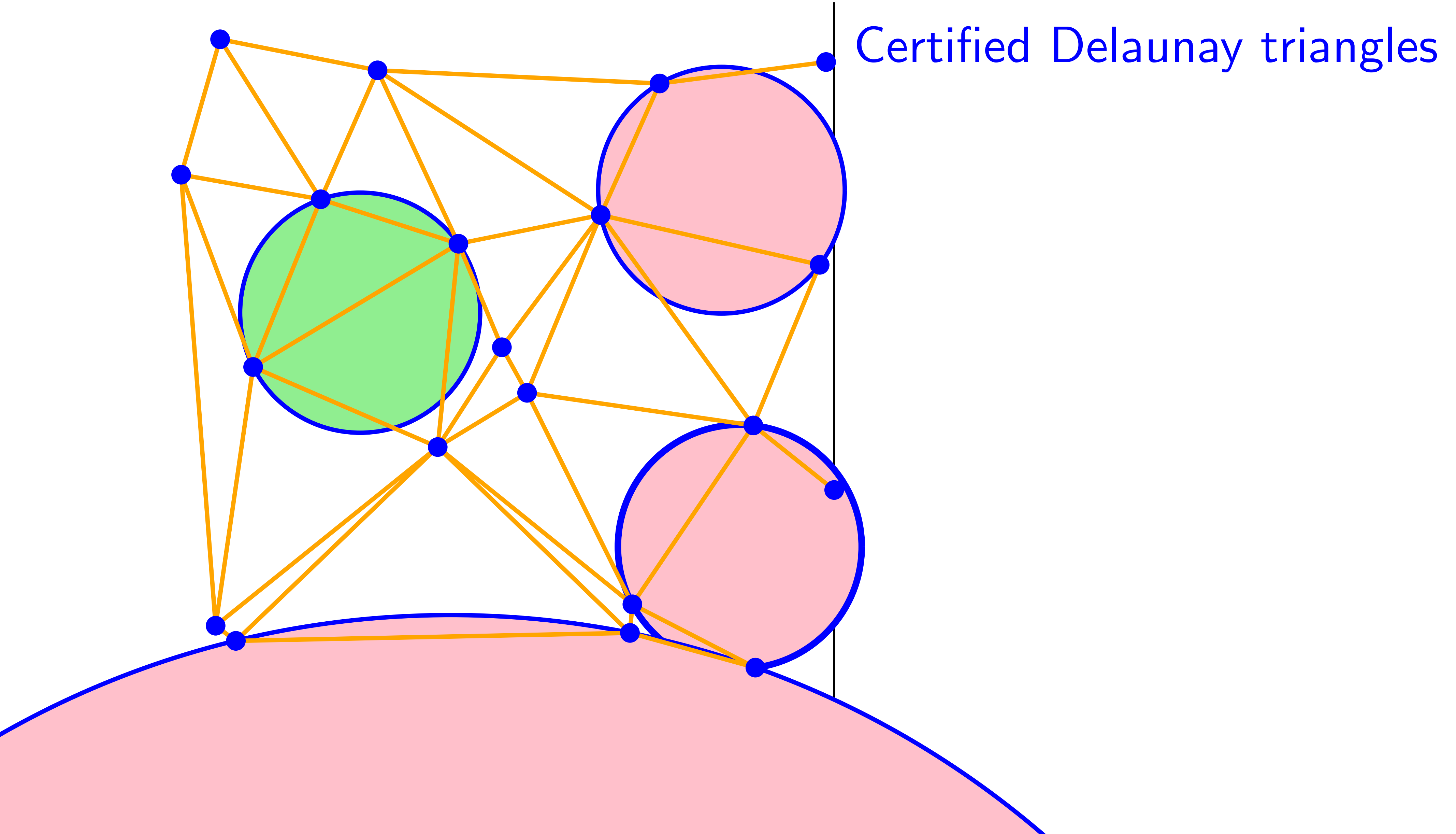
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



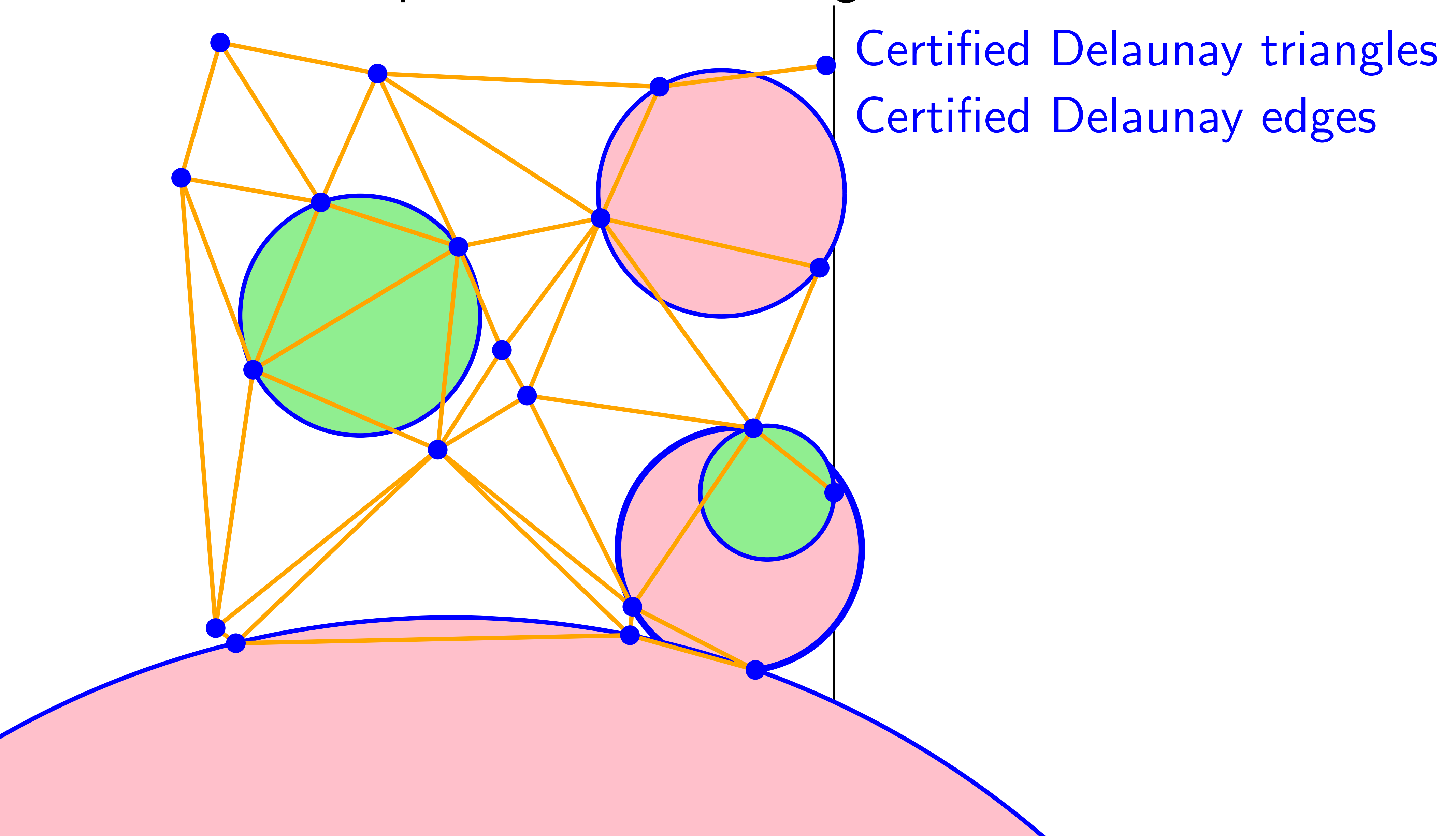
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



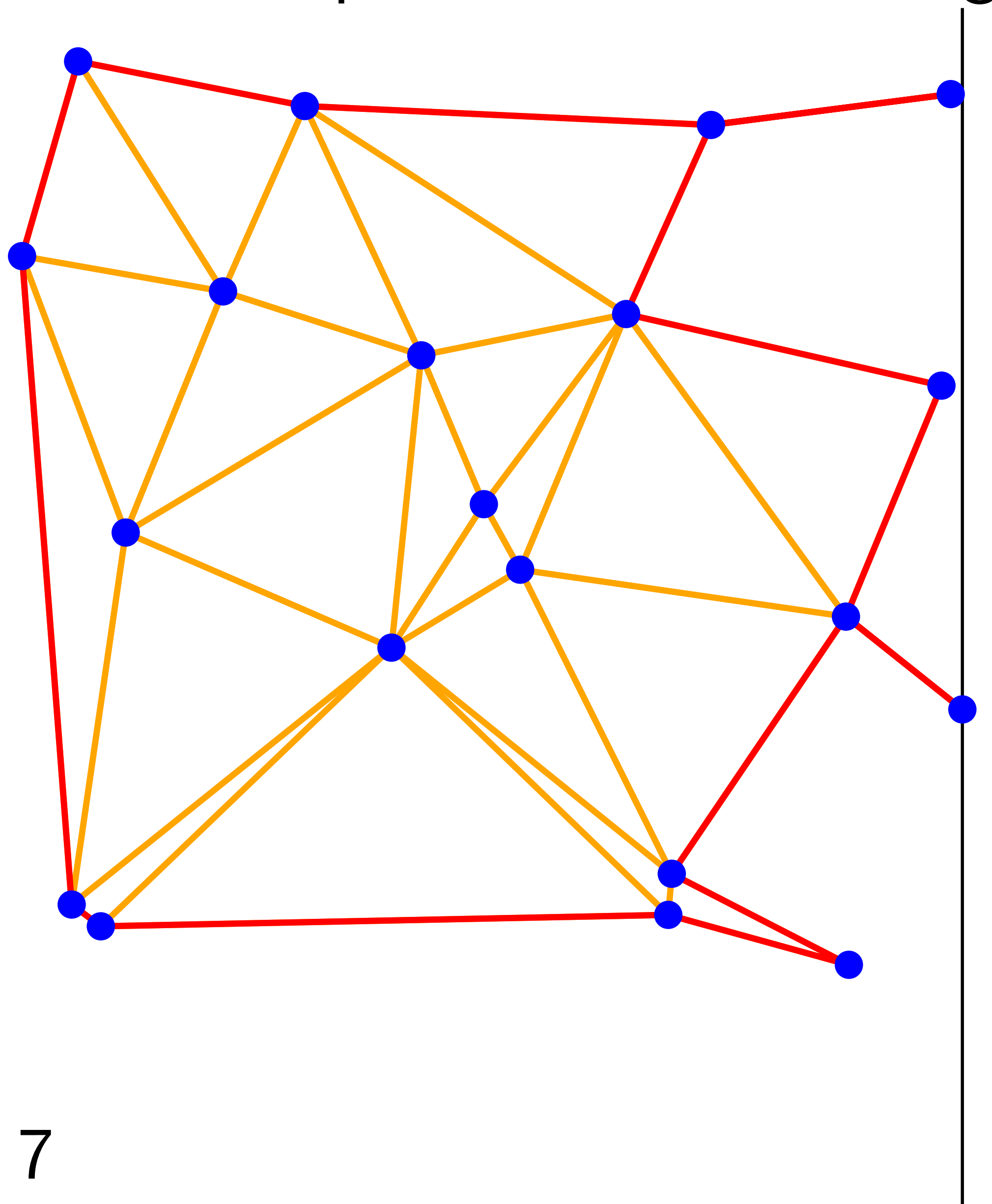
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



Delaunay Triangulation: sweep-line algorithm

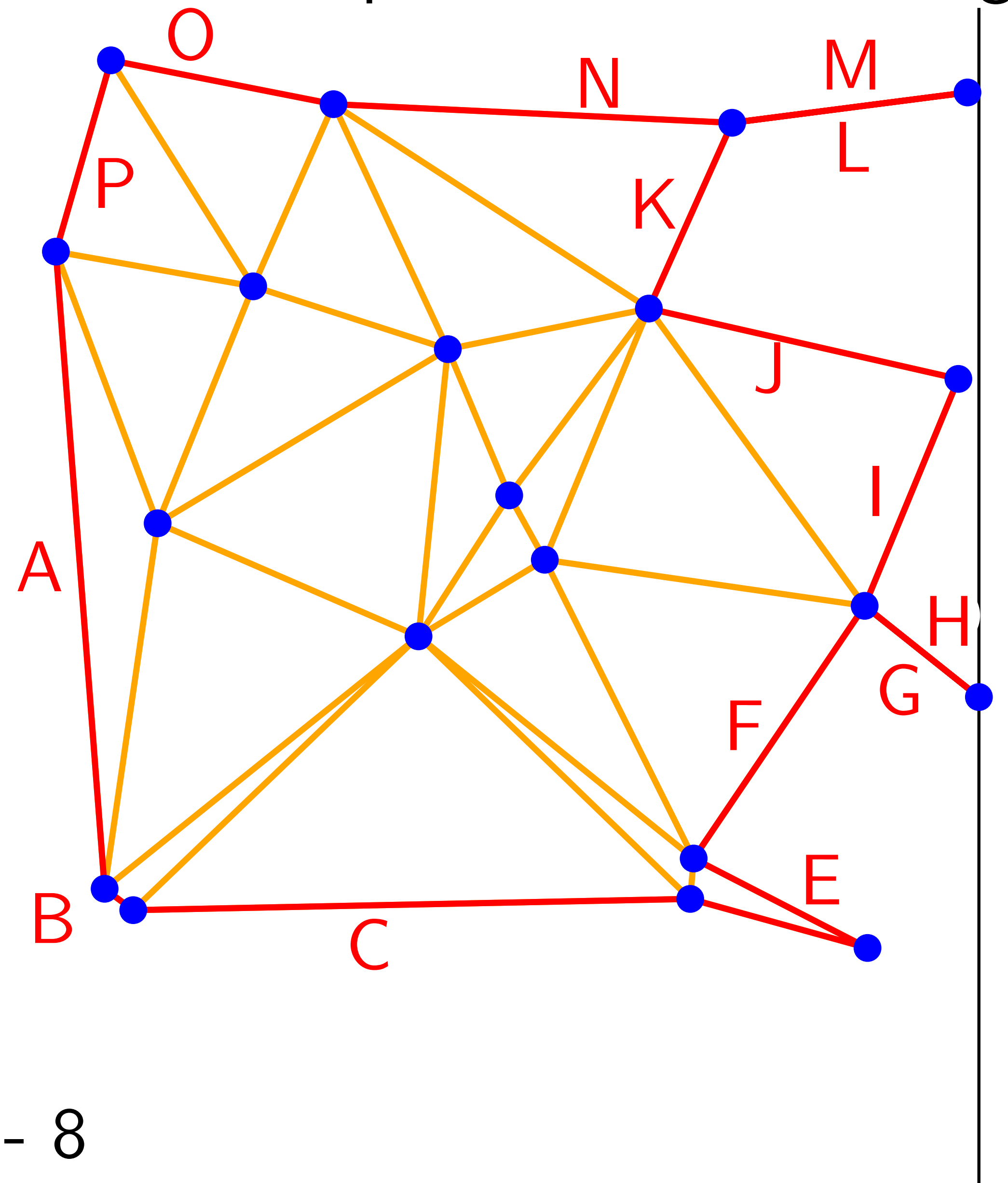
Discover the points from left to right



Boundary edges

Delaunay Triangulation: sweep-line algorithm

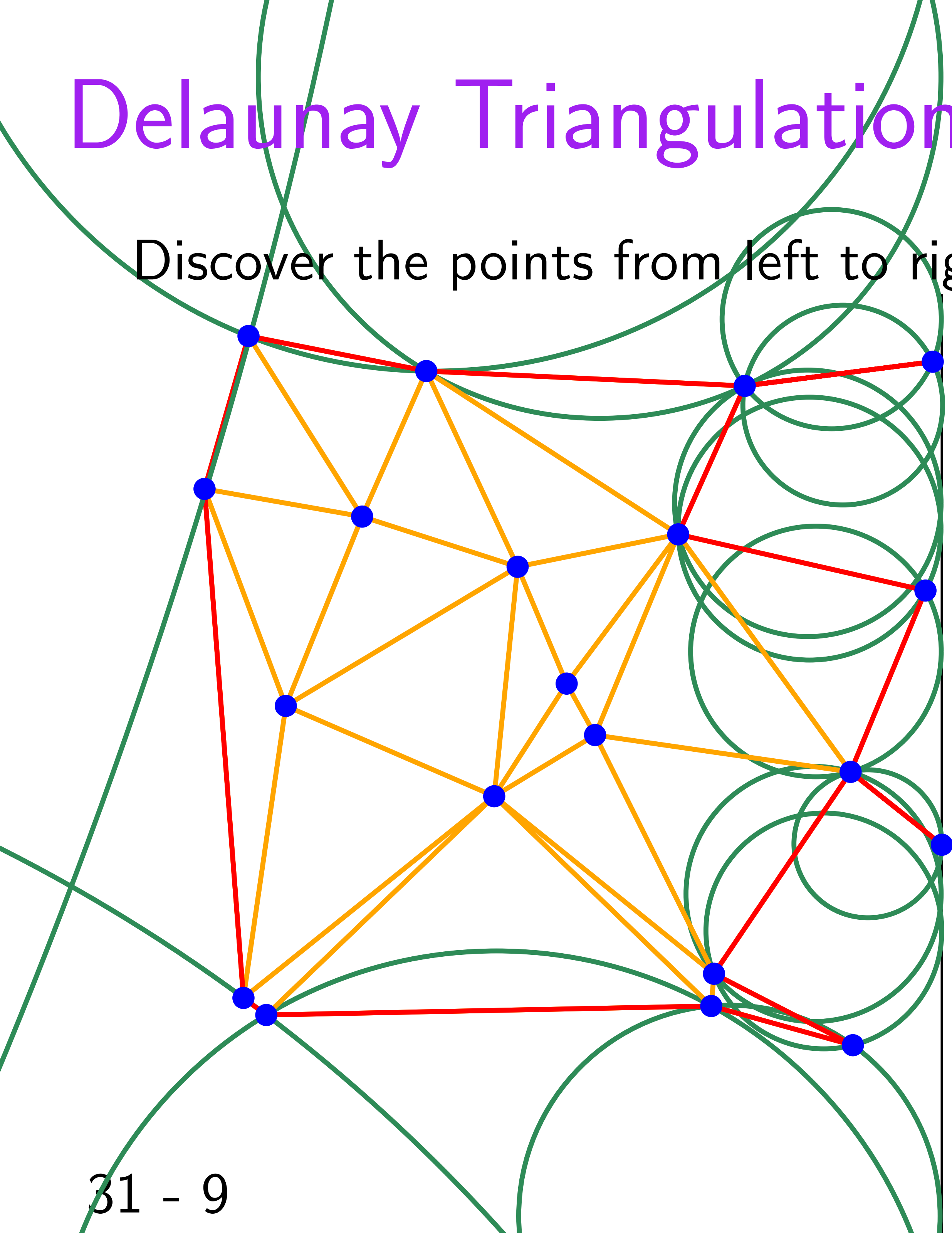
Discover the points from left to right



Boundary edges

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right

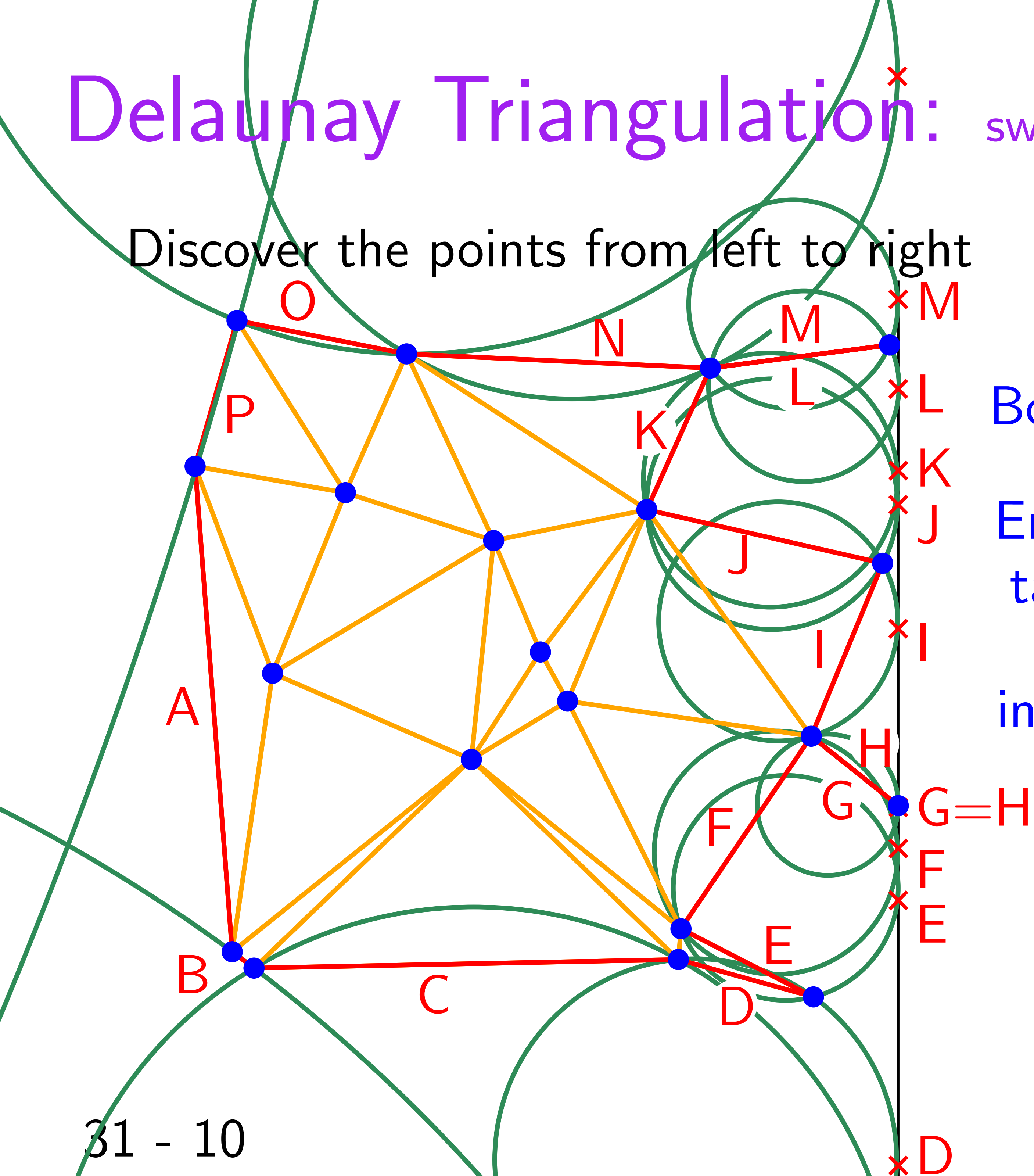


Boundary edges

Empty circles
tangent to sweep line

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



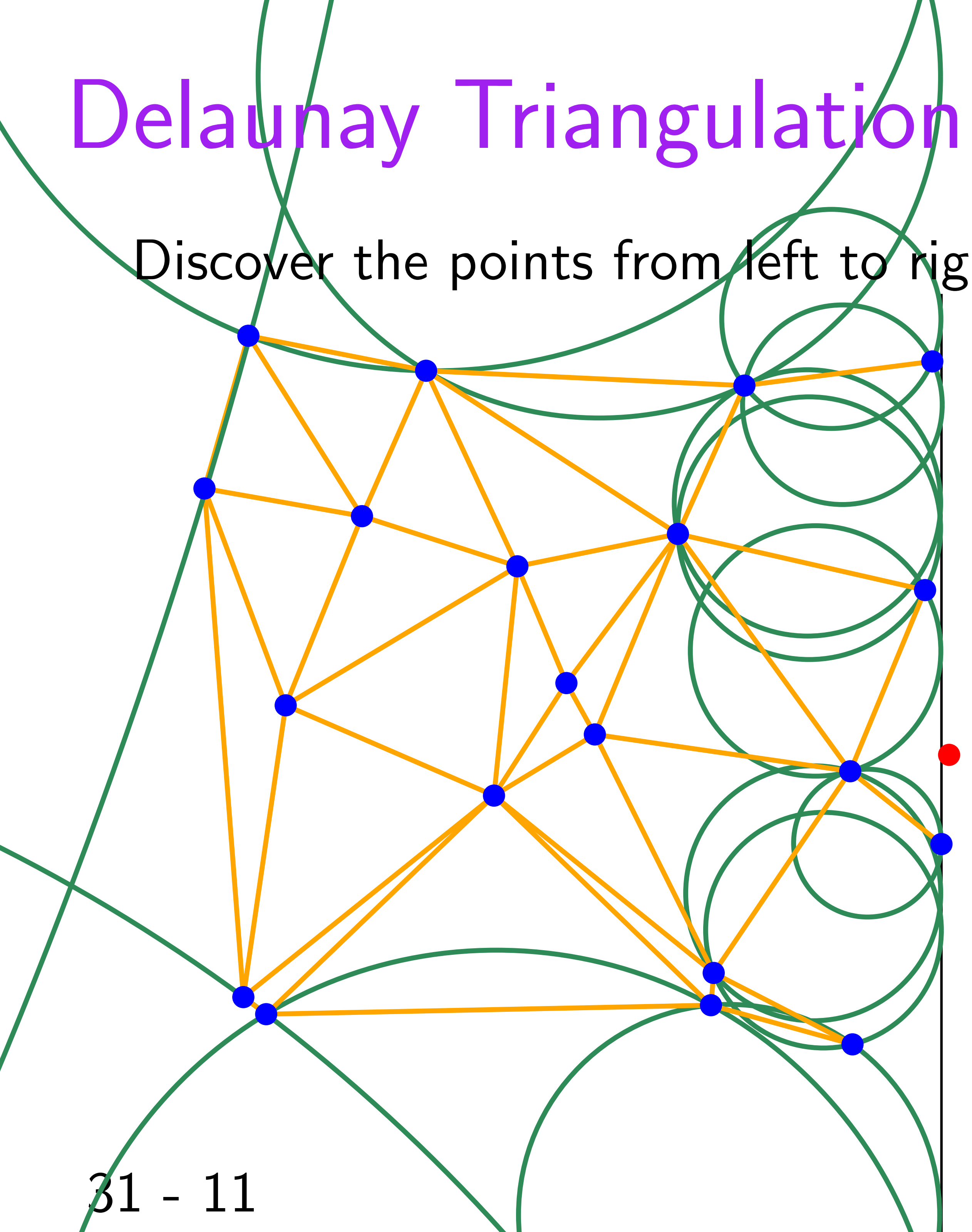
Boundary edges

Empty circles
tangent to sweep line

in order

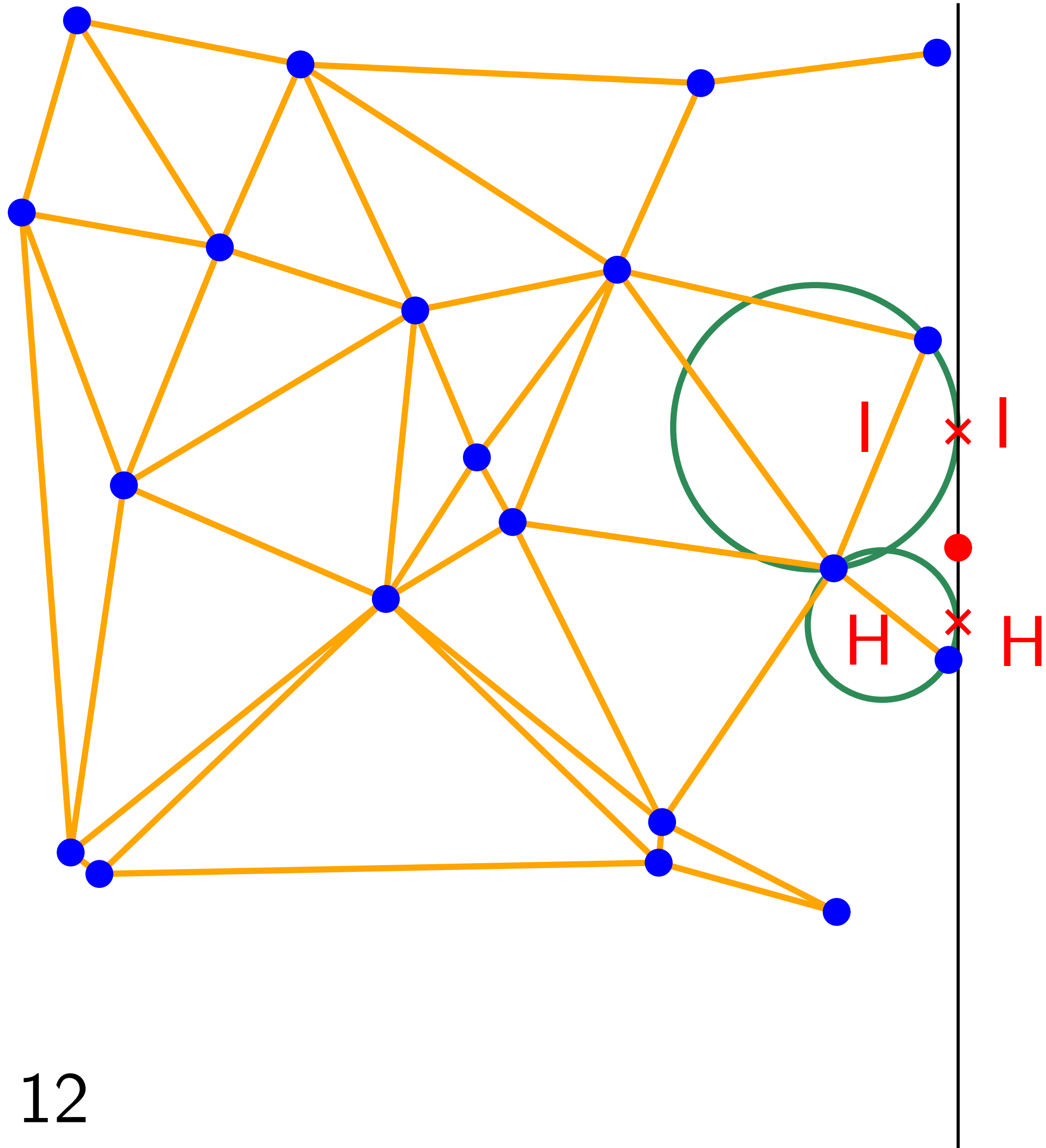
Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right

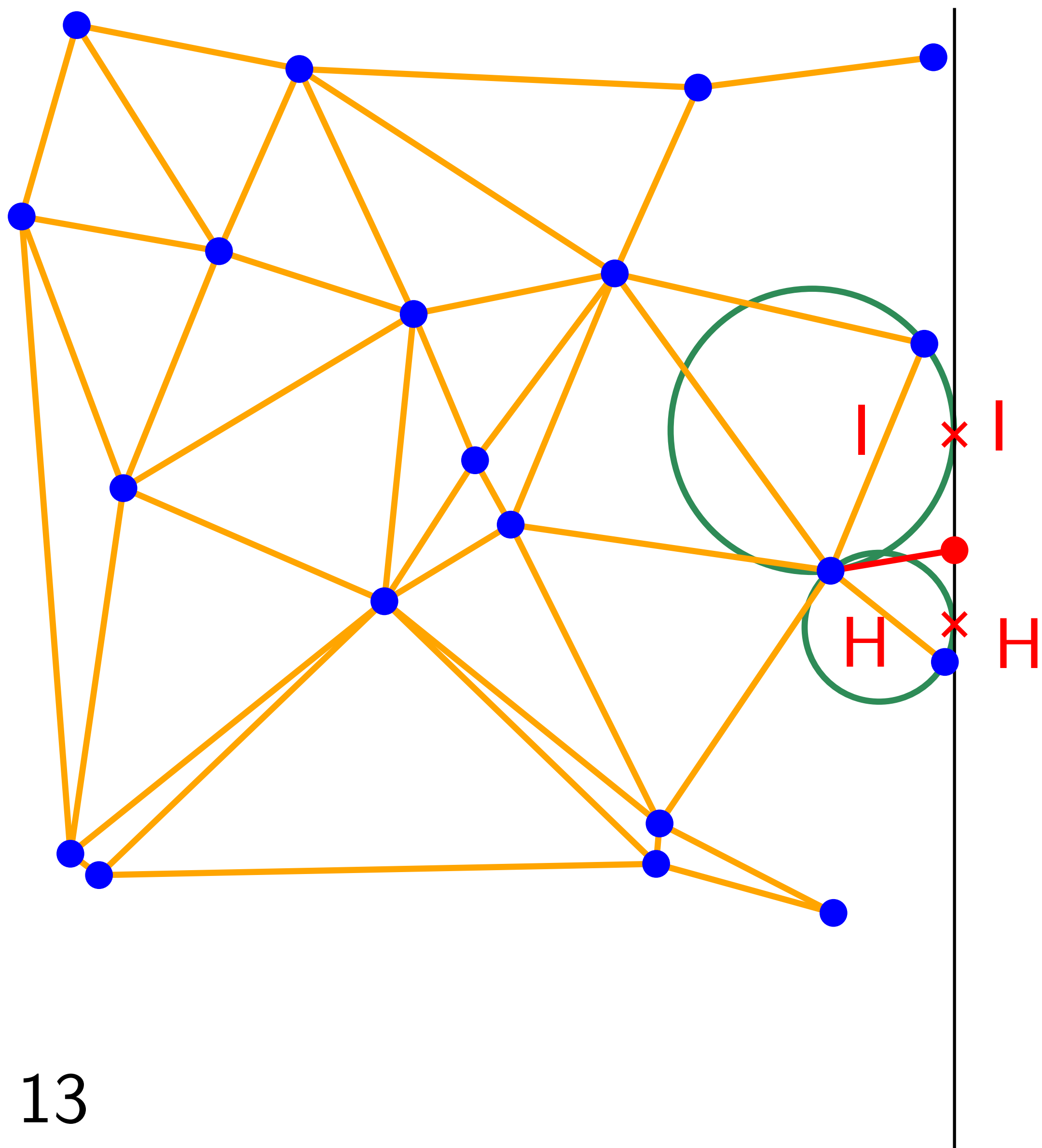


New point

Locate vertically

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



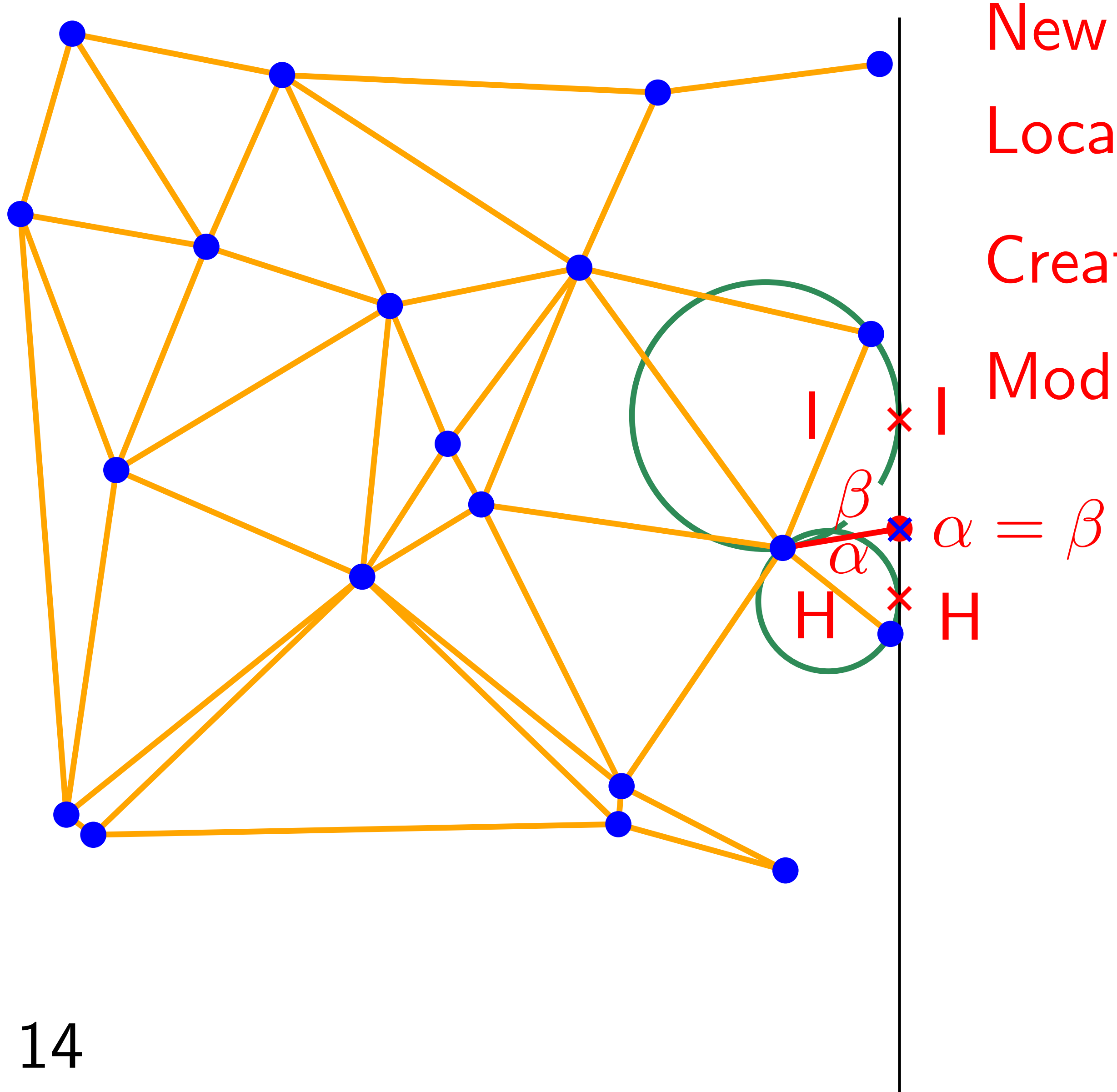
New point

Locate vertically

Create edge

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



New point

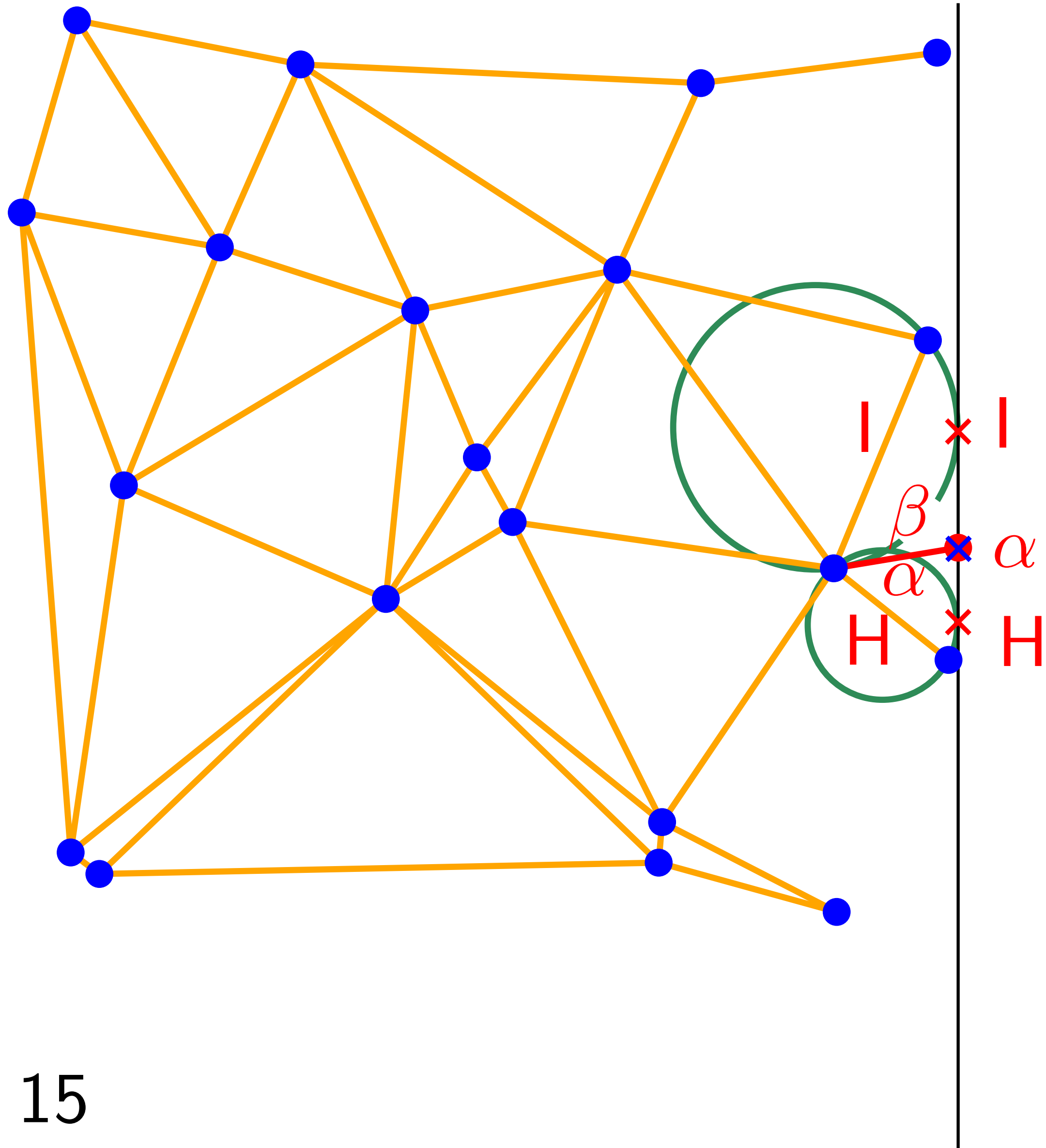
Locate vertically

Create edge

Modify boundary edges

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



New point

Locate vertically

Create edge

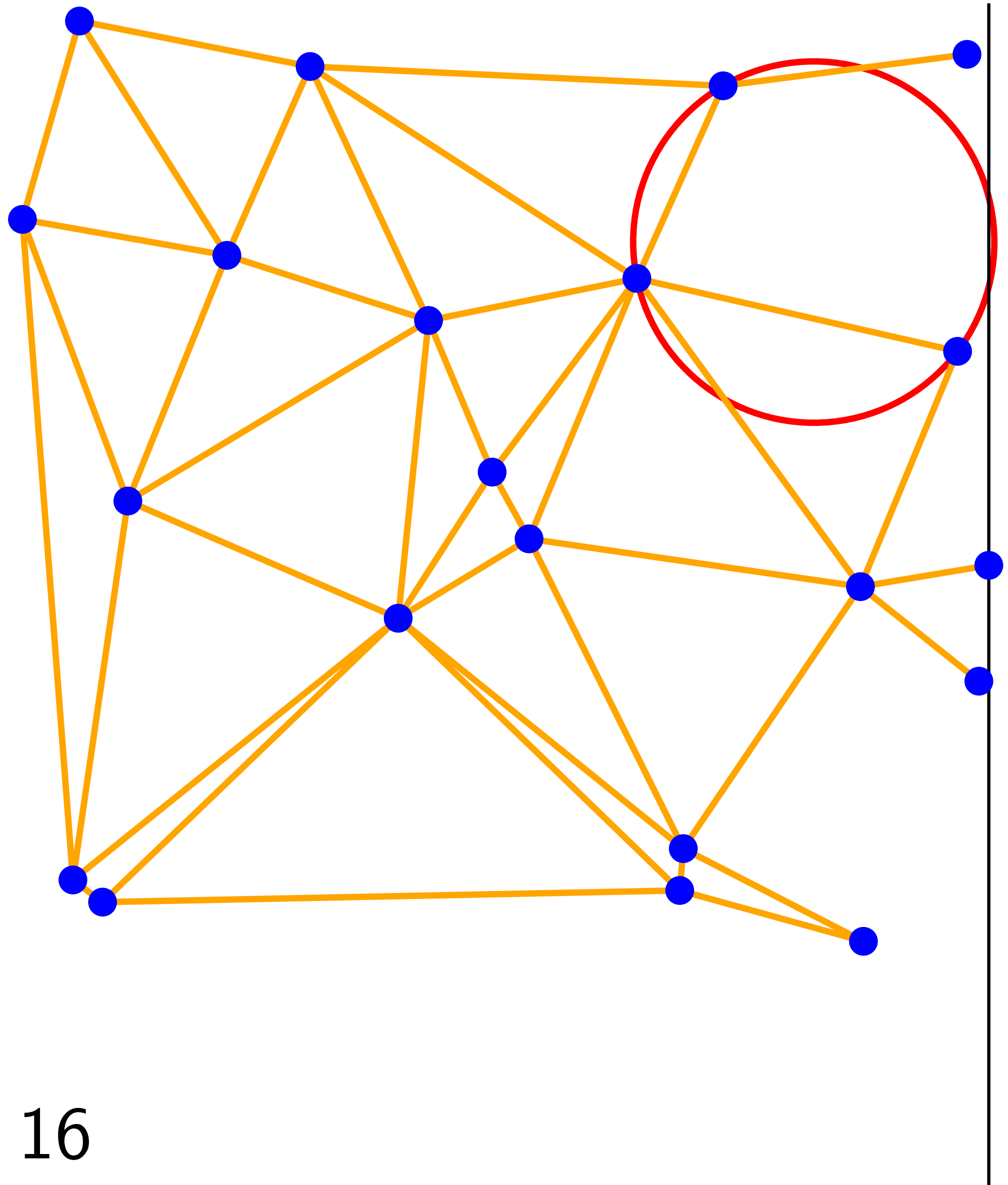
Modify boundary edges

Modify circle events

to be defined now

Delaunay Triangulation: sweep-line algorithm

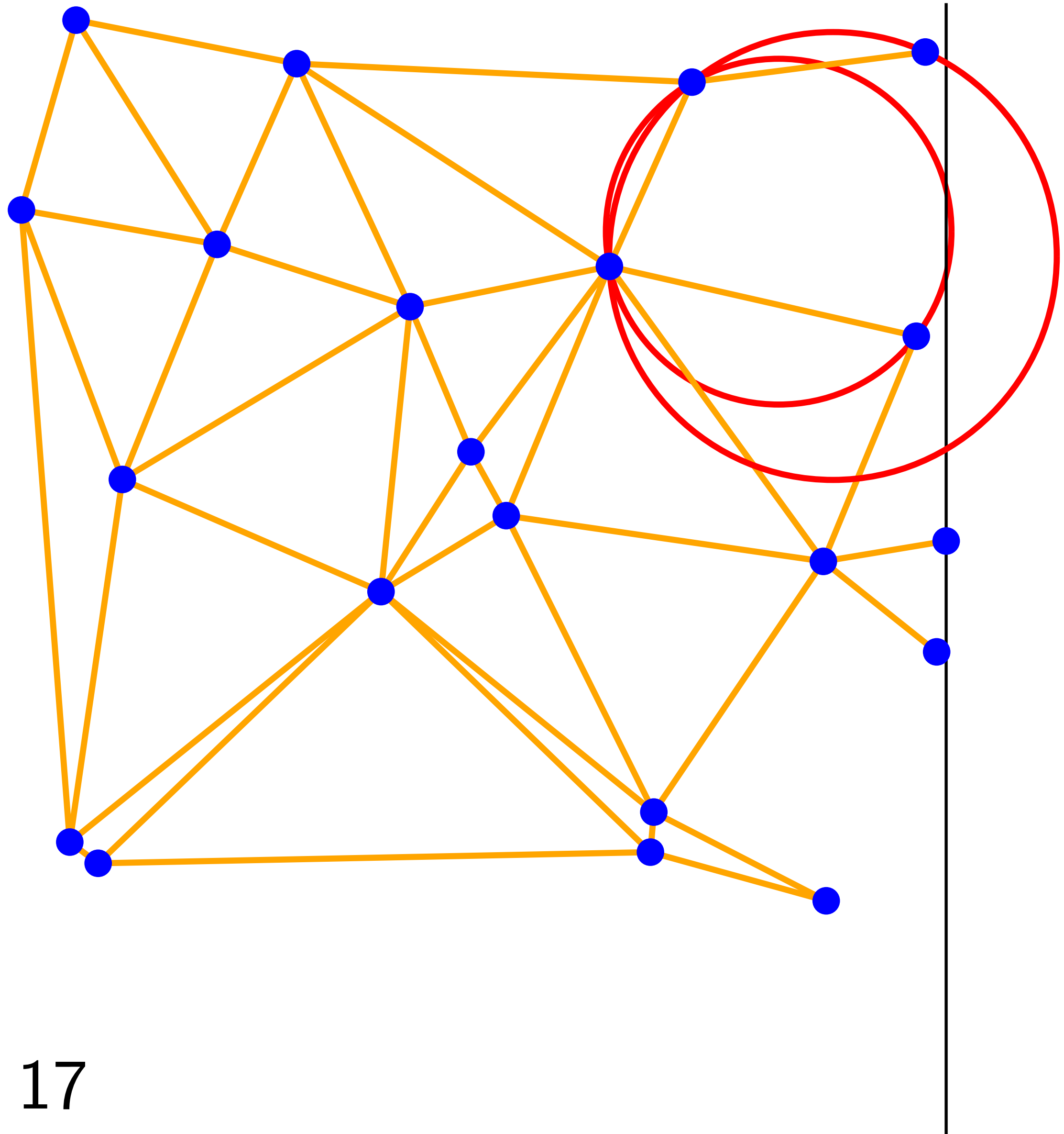
Discover the points from left to right



Closing a triangle ?

Delaunay Triangulation: sweep-line algorithm

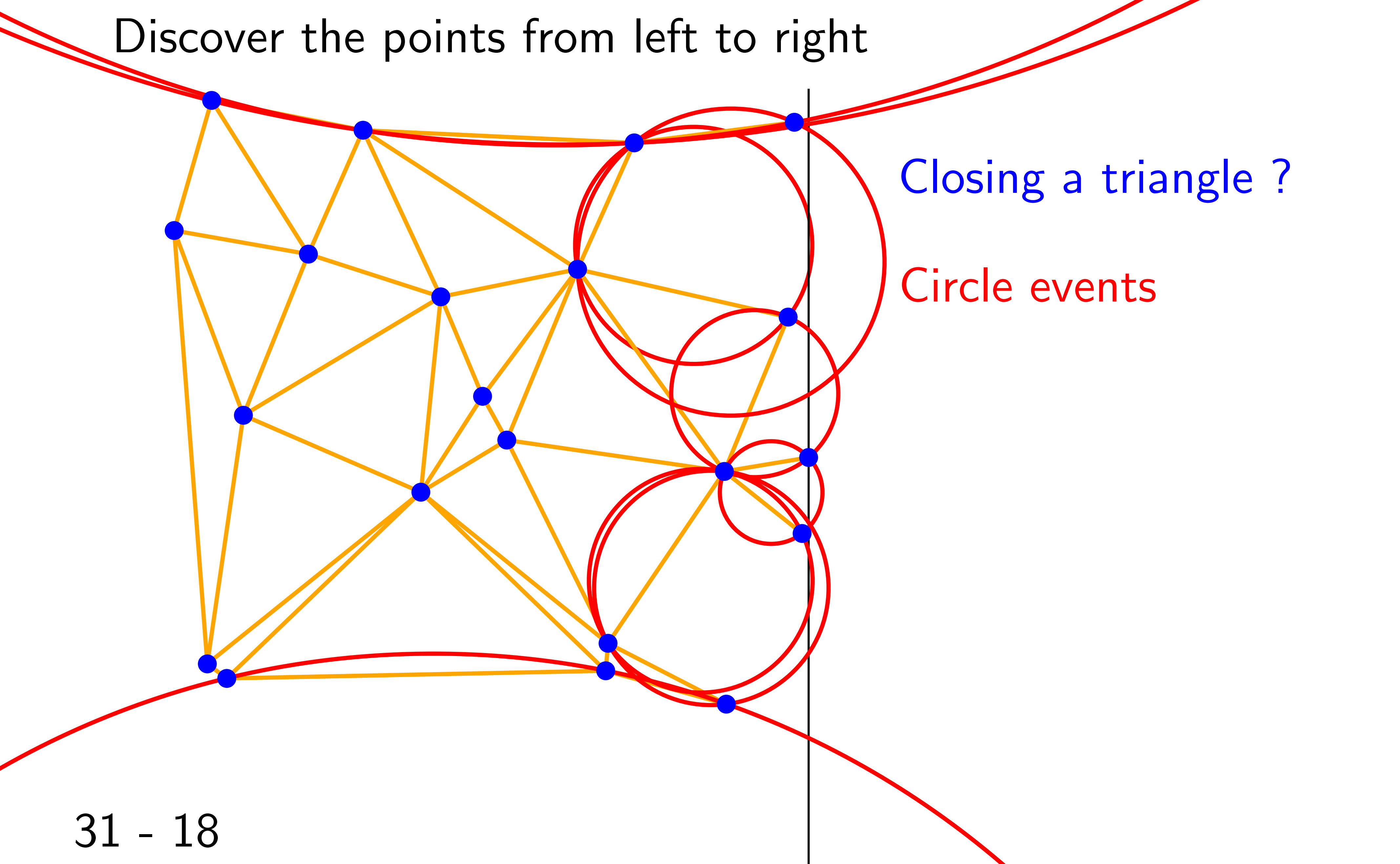
Discover the points from left to right



Closing a triangle ?

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right

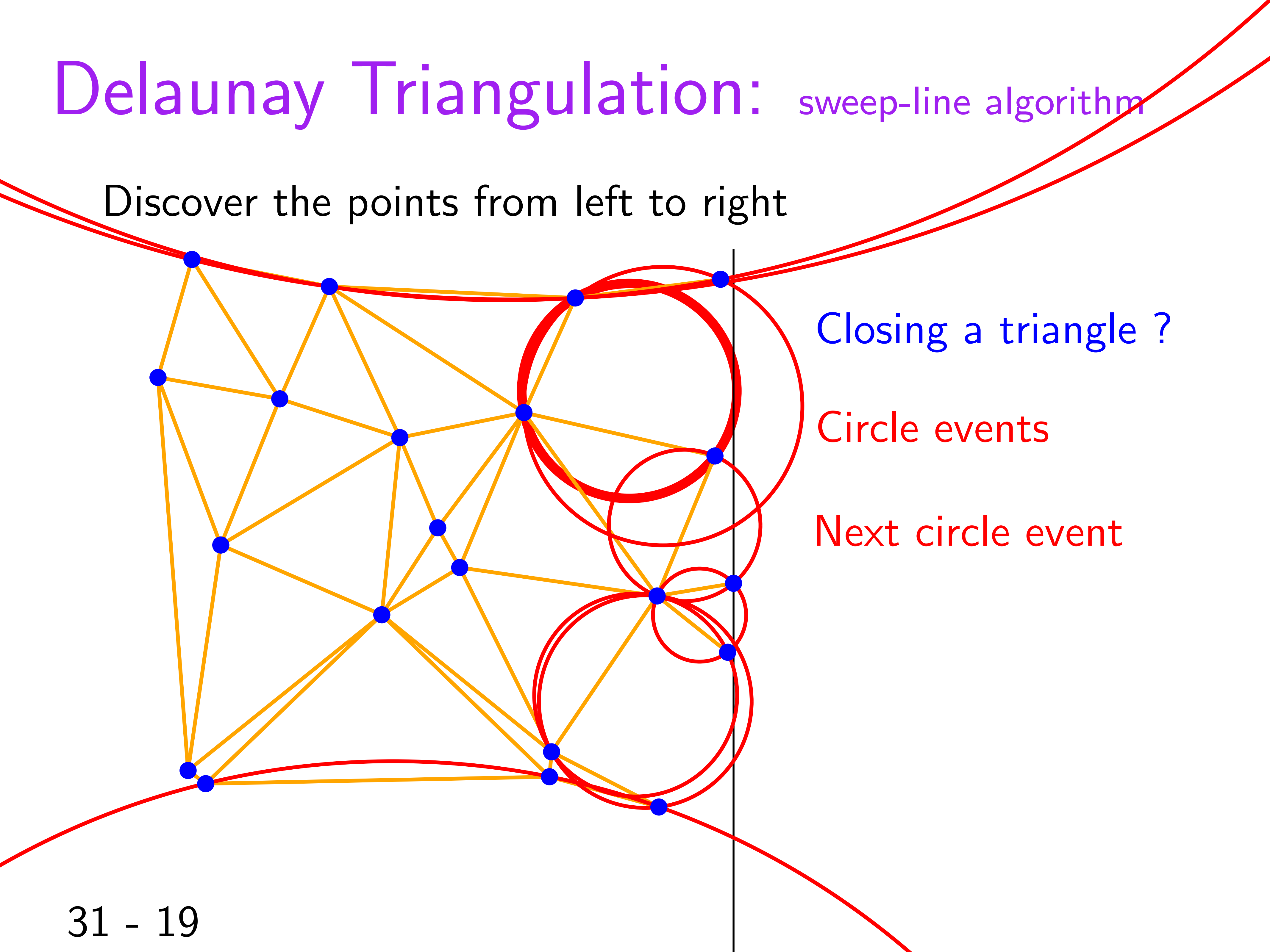


Closing a triangle ?

Circle events

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



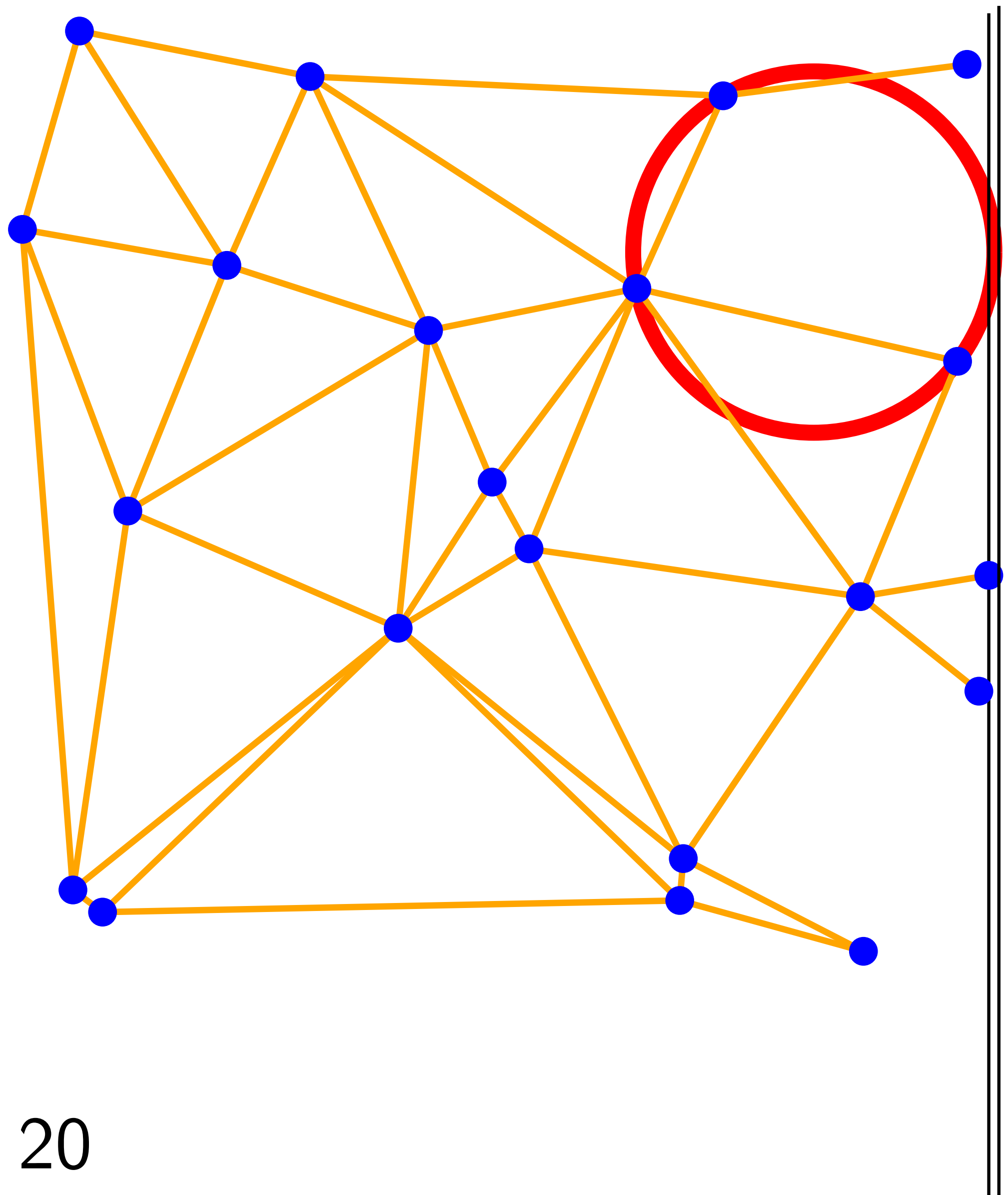
Closing a triangle ?

Circle events

Next circle event

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right

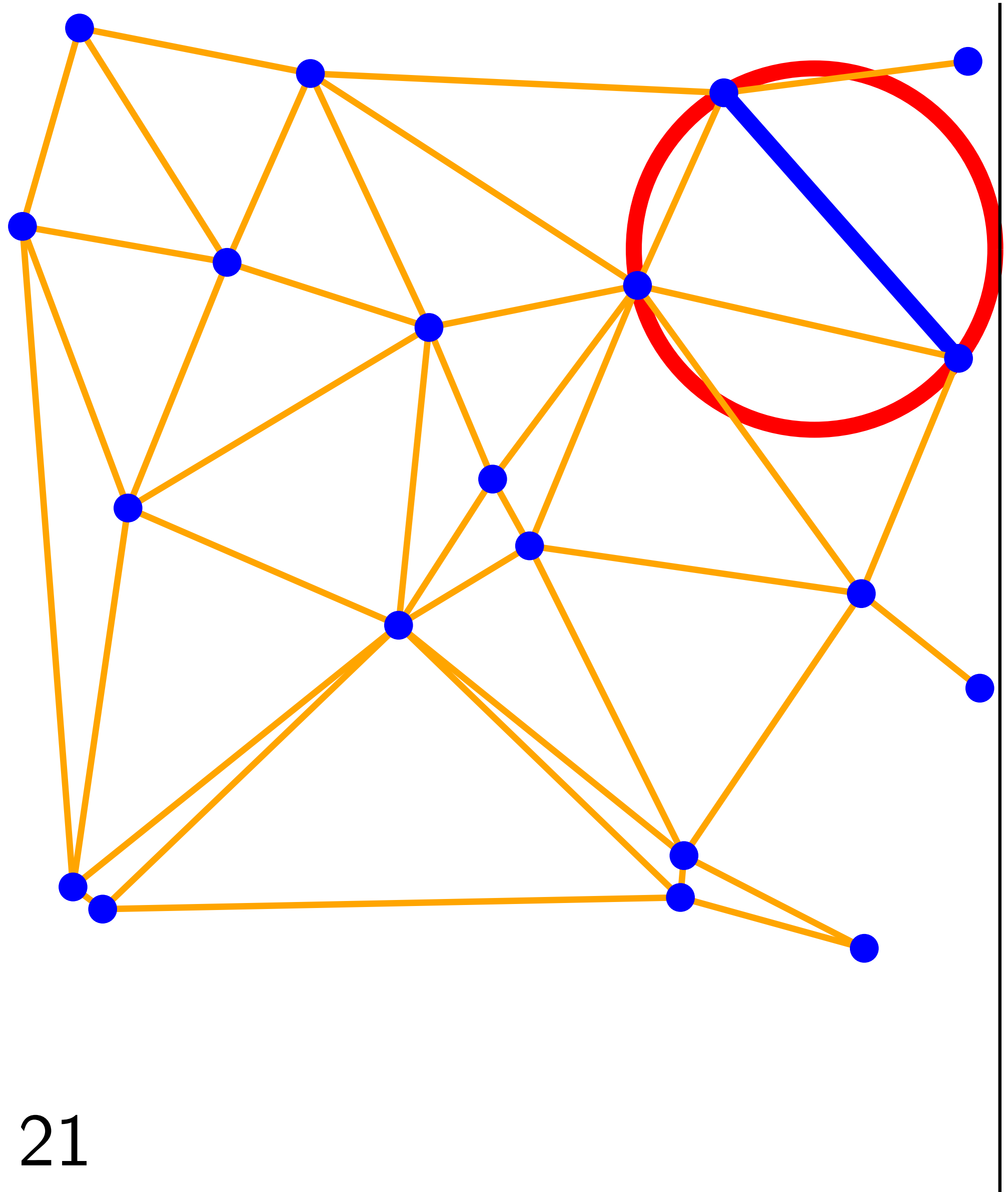


Closing a triangle ?

Next circle event

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right

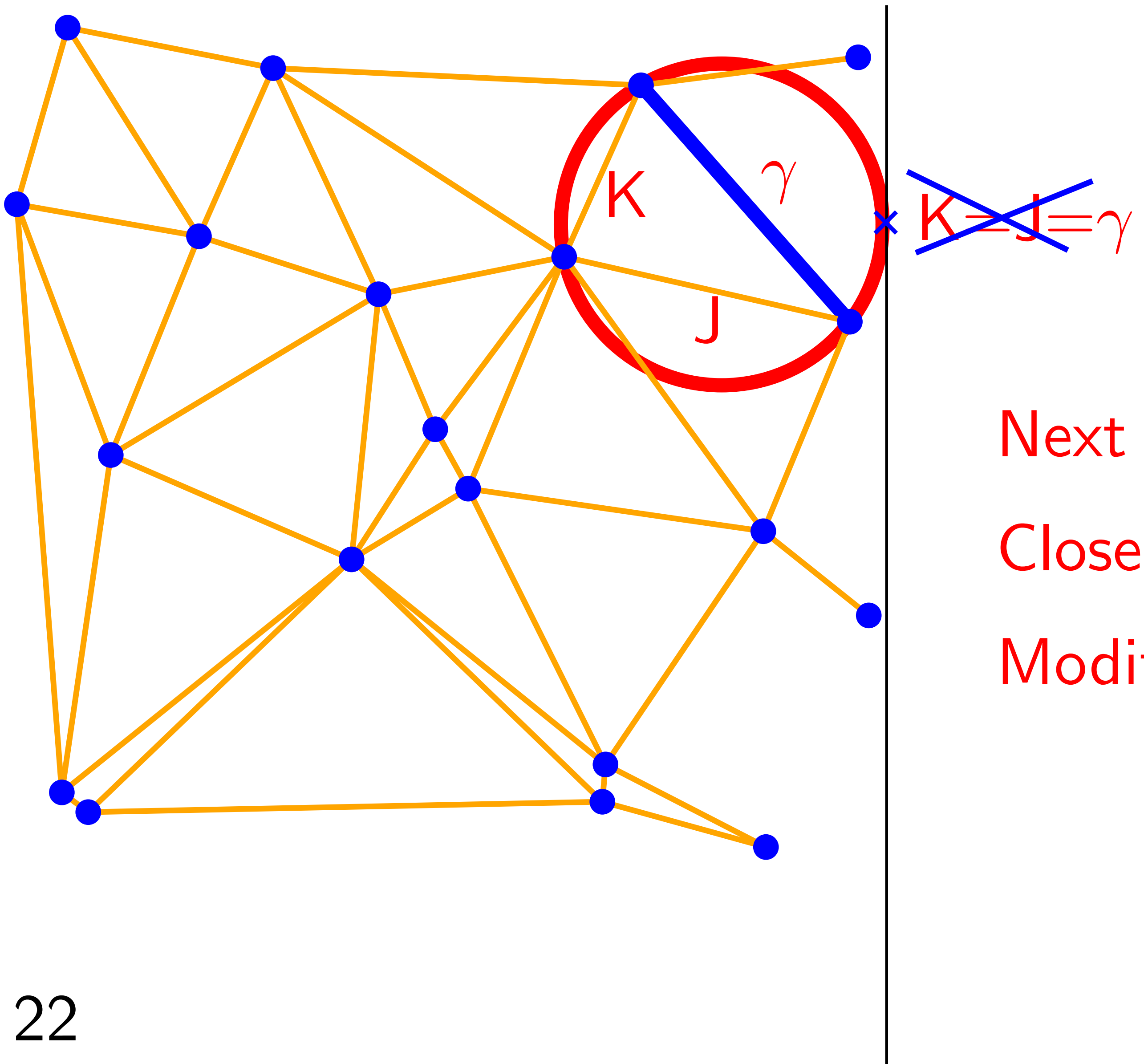


Next circle event

Close triangle

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



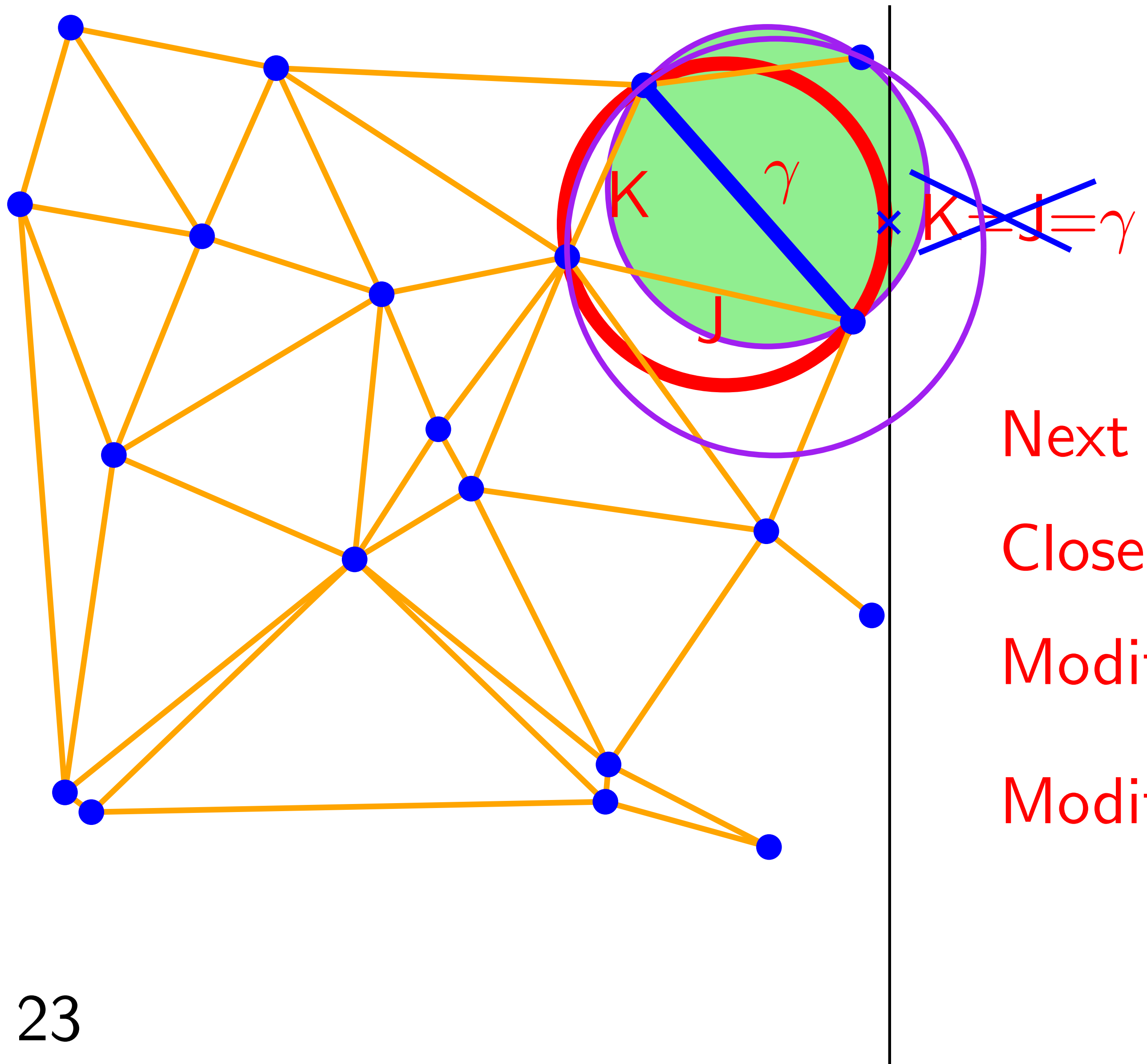
Next circle event

Close triangle

Modify boundary edges

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



Next circle event

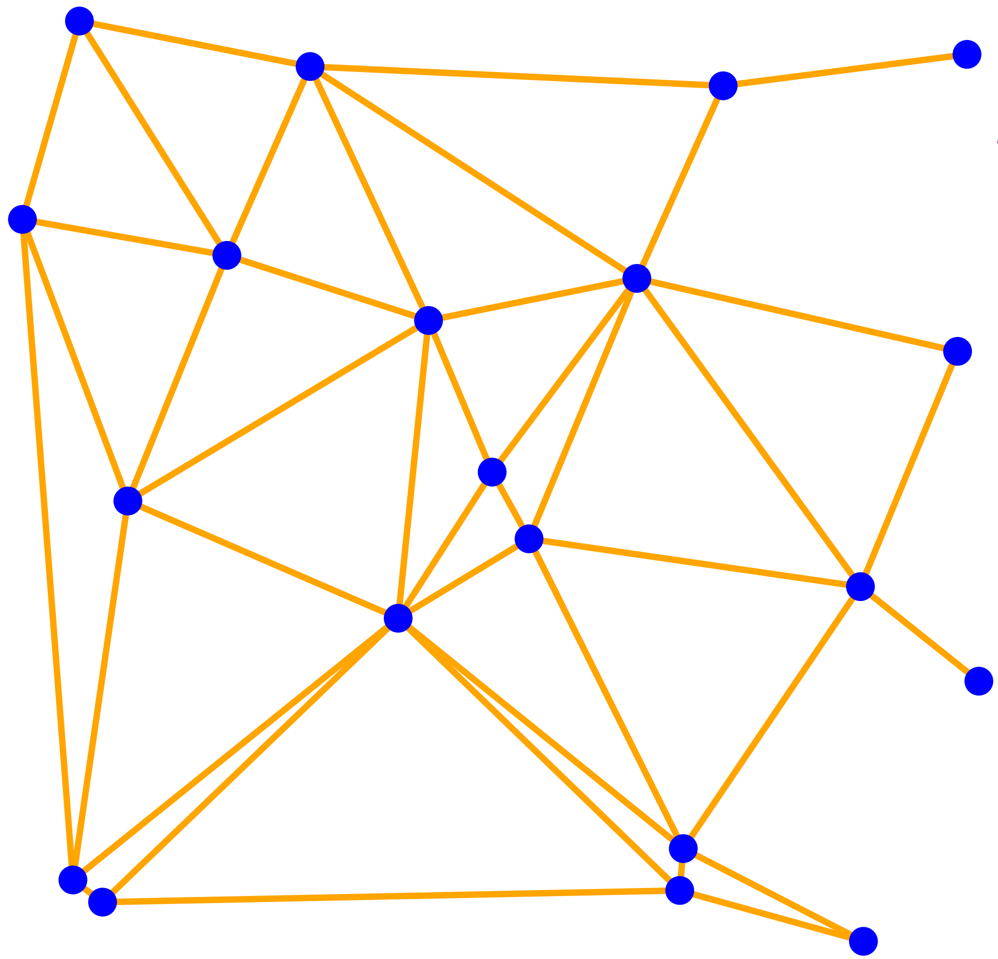
Close triangle

Modify boundary edges

Modify circle events

Delaunay Triangulation: sweep-line algorithm

Discover the points from left to right



Summary:

Process circle events
and point events
in x order

Three data structures
Triangulation
List of events (x sorted)
List of boundary edges
(ccw sorted)

Delaunay Triangulation: sweep-line algorithm

	Circle events	Point events
Complexity Number		
Triangulation		
List of events (x sorted)		
List of boundary edges (ccw sorted)		

Delaunay Triangulation: sweep-line algorithm

Complexity Number	Circle events processed	Point events
Triangulation		
List of events (x sorted)		
List of boundary edges (ccw sorted)		

Delaunay Triangulation: sweep-line algorithm

	Circle events processed	Point events
Complexity Number	$2n$	n
Triangulation		
List of events (x sorted)		
List of boundary edges (ccw sorted)		

Delaunay Triangulation: sweep-line algorithm

Complexity	Circle events processed	Point events
Number	$2n$	n
Triangulation	create 2 triangles per event	create one edge per event
List of events (x sorted)		
List of boundary edges (ccw sorted)		

Delaunay Triangulation: sweep-line algorithm

Complexity	Circle events processed	Point events
Number	$2n$	n
Triangulation	create 2 triangles per event	create one edge per event
List of events (x sorted)	≤ 3 deletions ≤ 2 insertions per event	≤ 2 deletions ≤ 2 insertions per event
List of boundary edges (ccw sorted)		

Delaunay Triangulation: sweep-line algorithm

Complexity	Circle events processed	Point events
Number	$2n$	n
Triangulation	create 2 triangles per event	create one edge per event
List of events (x sorted)	≤ 3 deletions ≤ 2 insertions per event	≤ 2 deletions ≤ 2 insertions per event
List of boundary edges (ccw sorted)	replace 2 edges by 1 per event	locate, then insert 2 edges per event

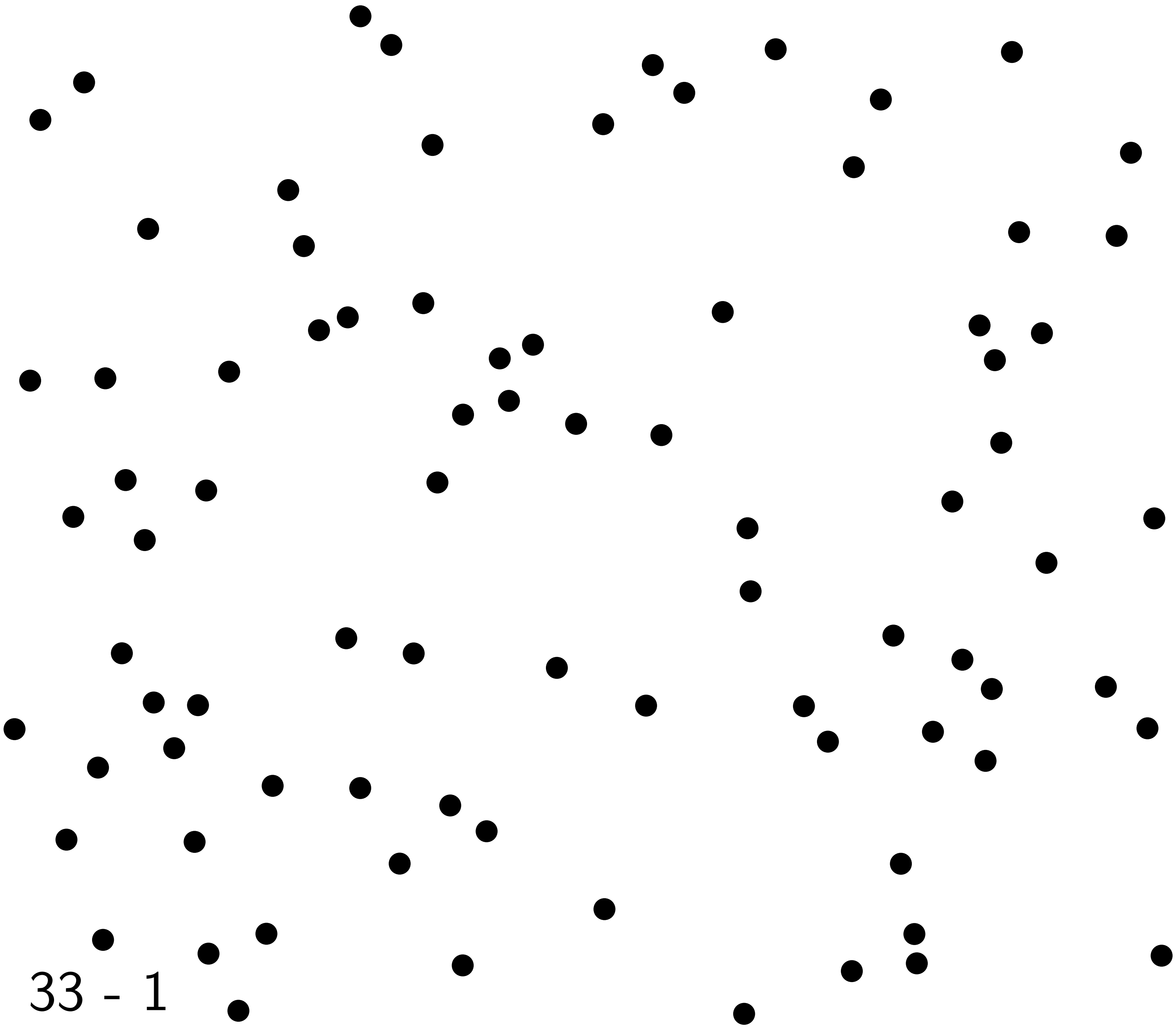
Delaunay Triangulation: sweep-line algorithm

	Circle events	Point events
Complexity	processed	
Number	$2n$	n
Triangulation $O(1)$ per operation	create 2 triangles per event	create one edge per event
List of events (x sorted) $O(\log n)$ per operation	≤ 3 deletions ≤ 2 insertions per event	≤ 2 deletions ≤ 2 insertions per event
List of boundary edges (ccw sorted) $O(\log n)$ per operation	replace 2 edges by 1 per event	locate, then insert 2 edges per event

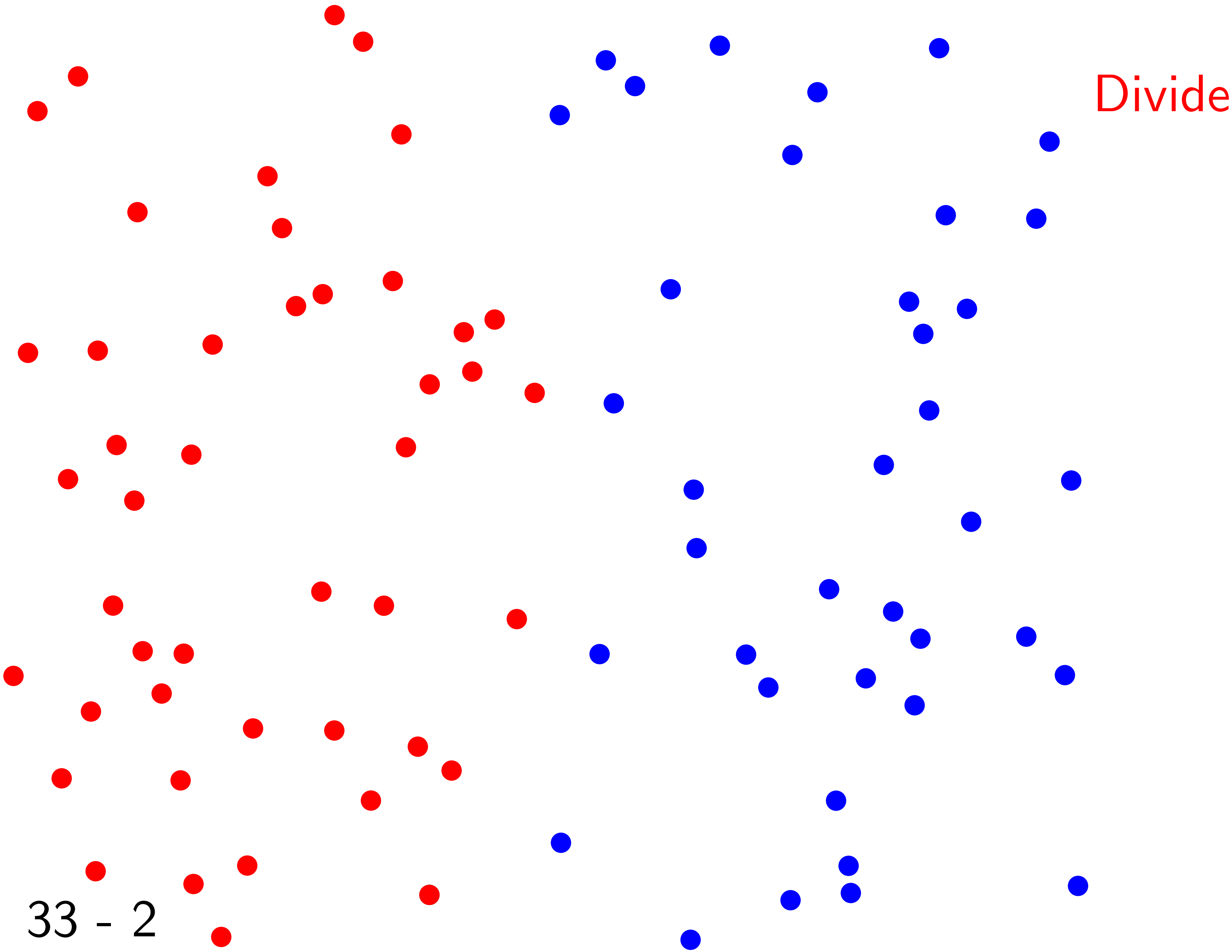
Delaunay Triangulation: sweep-line algorithm

Complexity Number	Circle events processed $2n$	Point events n
Triangulation $O(1)$ per operation	create 2 triangles $O(n \log n)$	create one edge
List of events (x sorted) $O(\log n)$ per operation	per event	per event
List of boundary edges (ccw sorted) $O(\log n)$ per operation	replace 2 edges by 1 per event	locate, then insert 2 edges per event

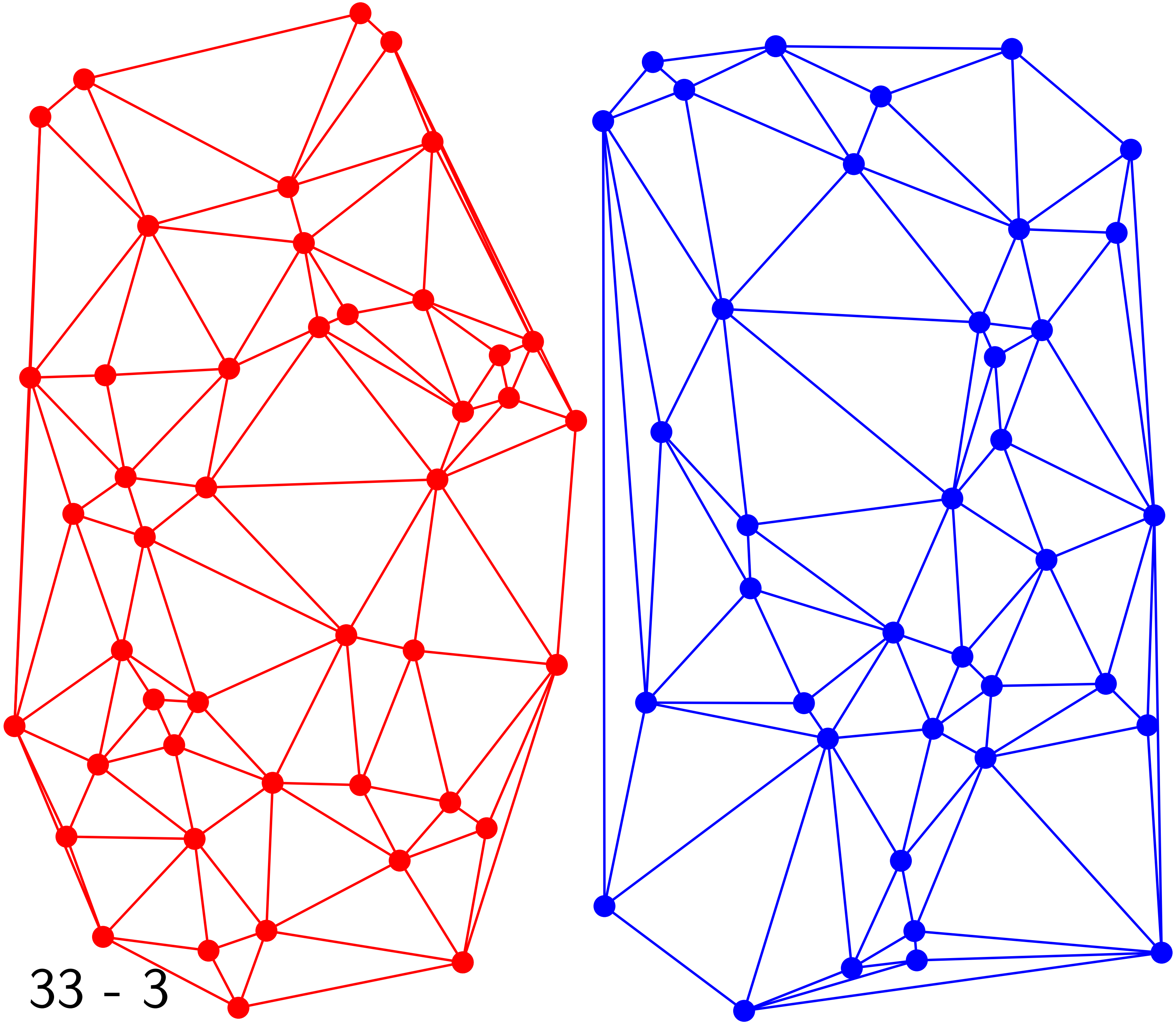
Delaunay Triangulation: divide & conquer (sketch)



Delaunay Triangulation: divide & conquer (sketch)



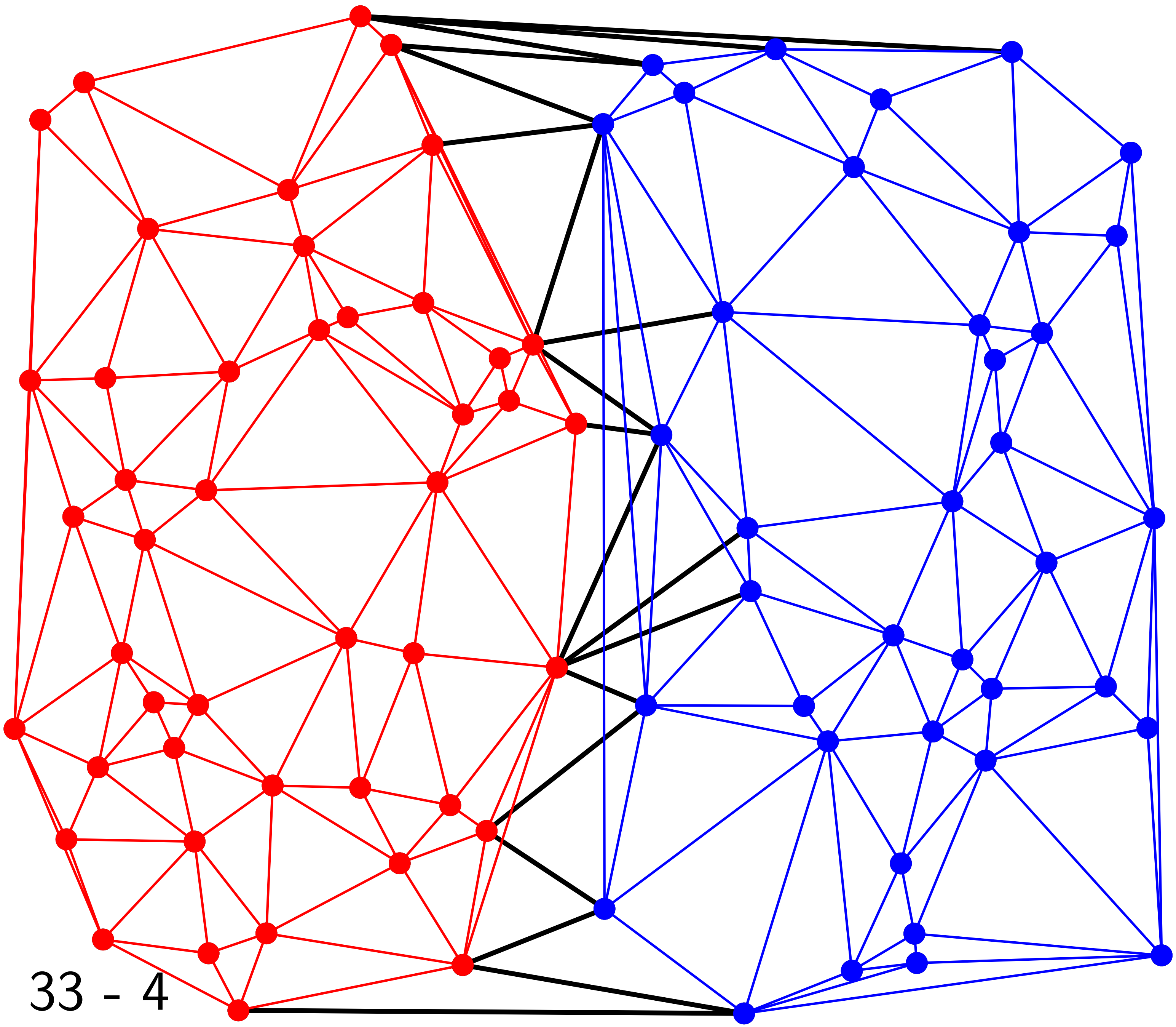
Delaunay Triangulation: divide & conquer (sketch)



Divide

Recurse

Delaunay Triangulation: divide & conquer (sketch)

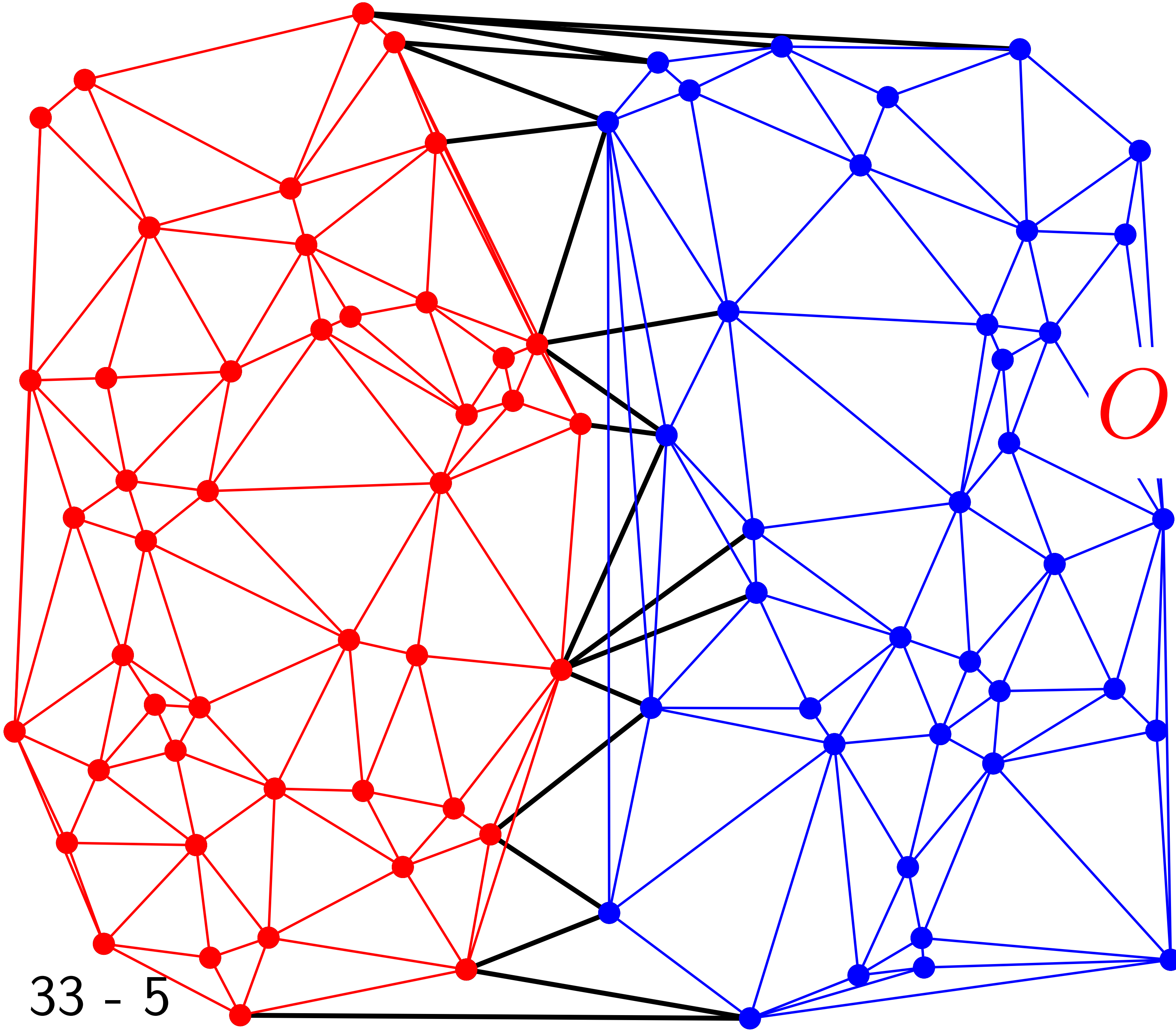


Divide

Recurse

Conquer

Delaunay Triangulation: divide & conquer (sketch)



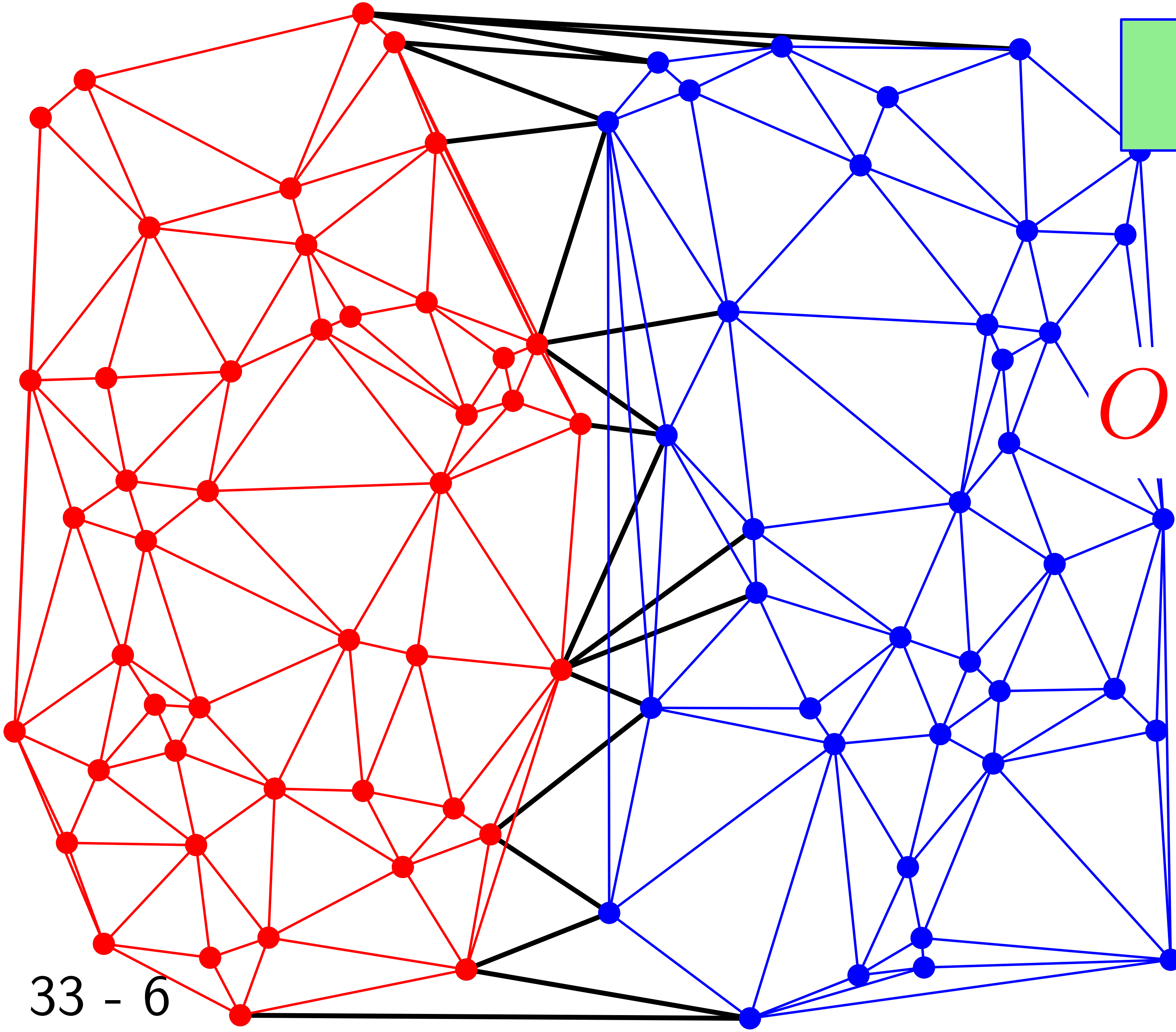
Divide

Recurse

Conquer

$O(n \log n)$

Delaunay Triangulation: divide & conquer (sketch)



Divide

Recurse

Conquer

$O(n \log n)$

balanced

linear time

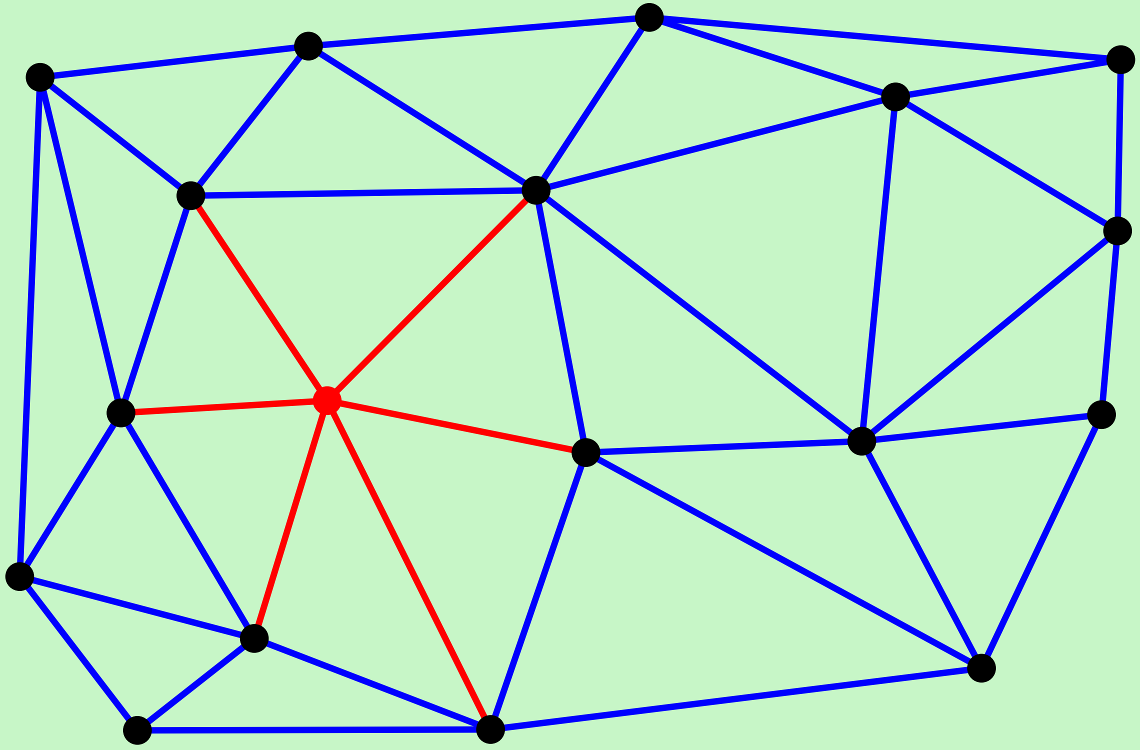
easier conquer

Delaunay Triangulation: deletion algorithm (sketch)

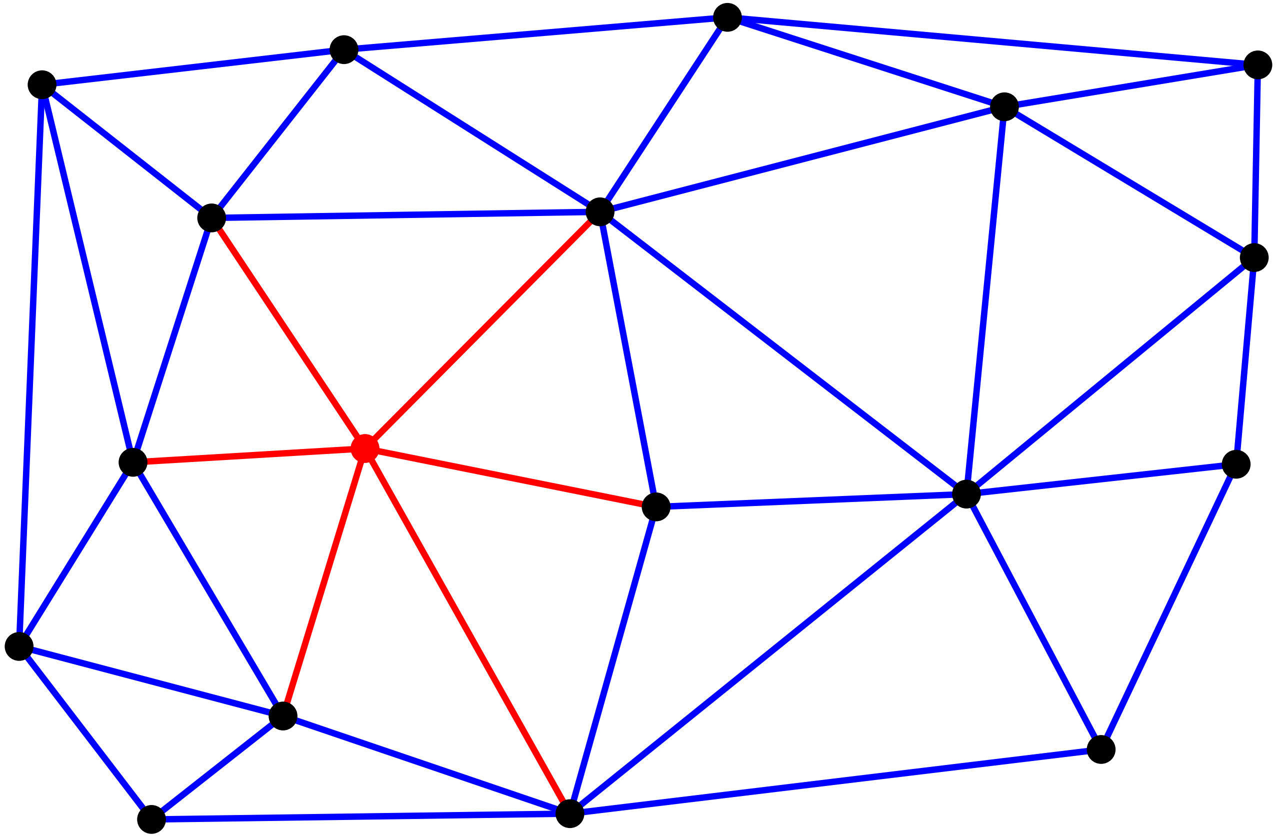
Delaunay Triangulation: deletion algorithm (sketch)

Delaunay Triangulation: incremental algorithm

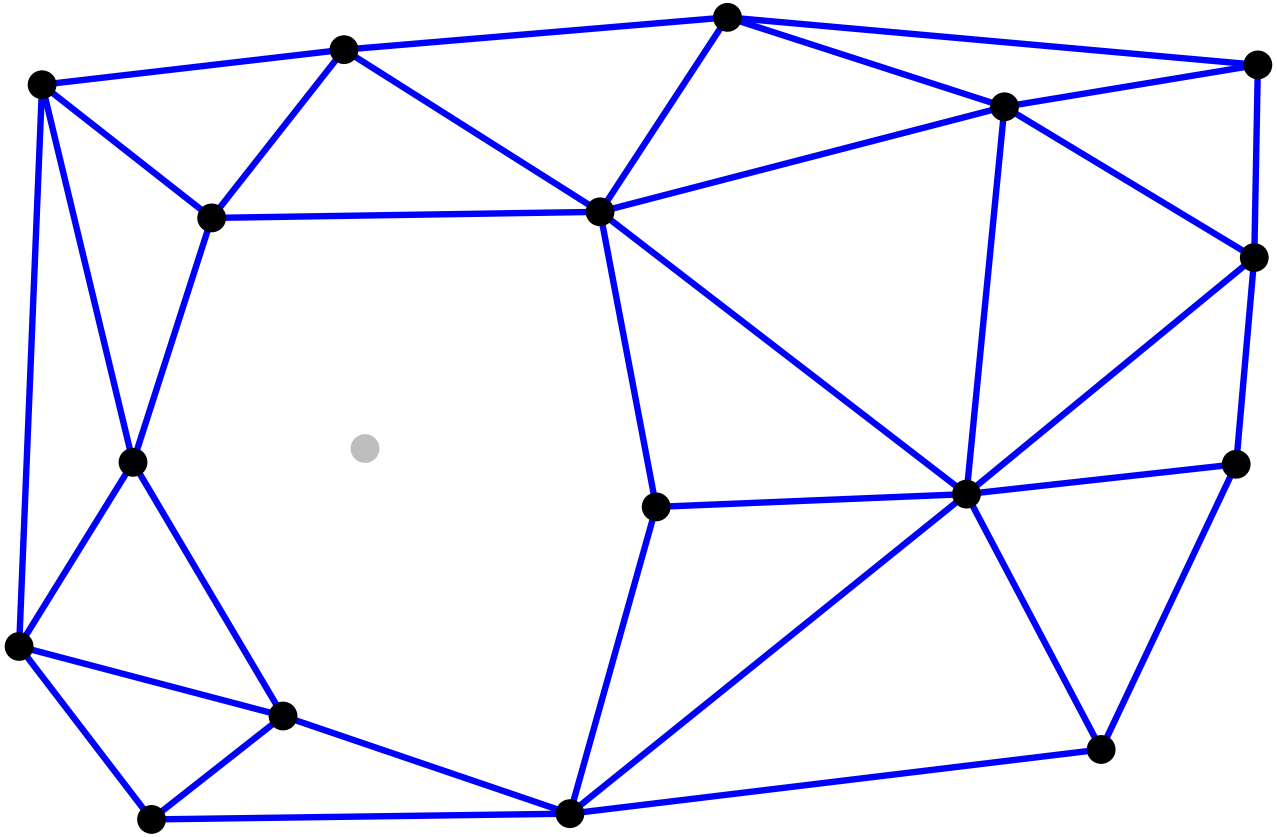
New point



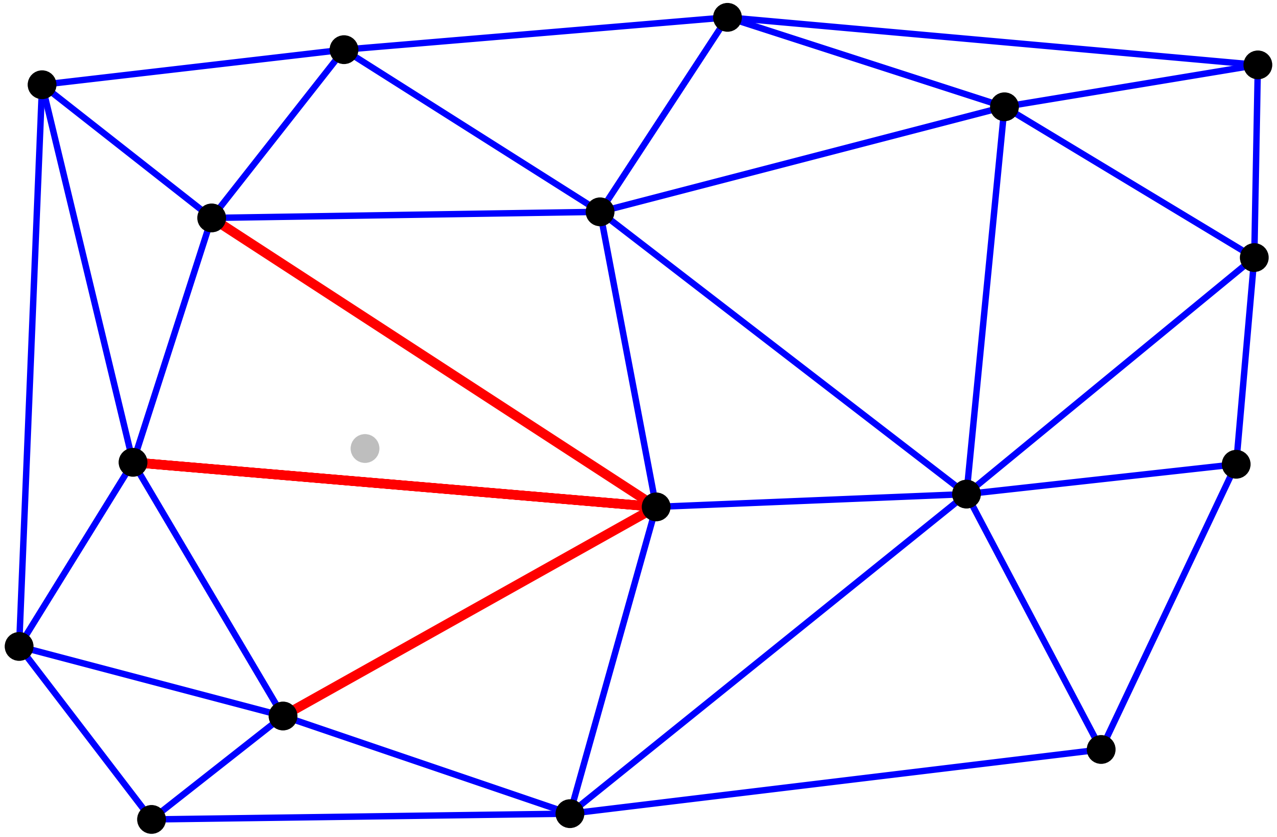
Delaunay Triangulation: deletion algorithm (sketch)



Delaunay Triangulation: deletion algorithm (sketch)

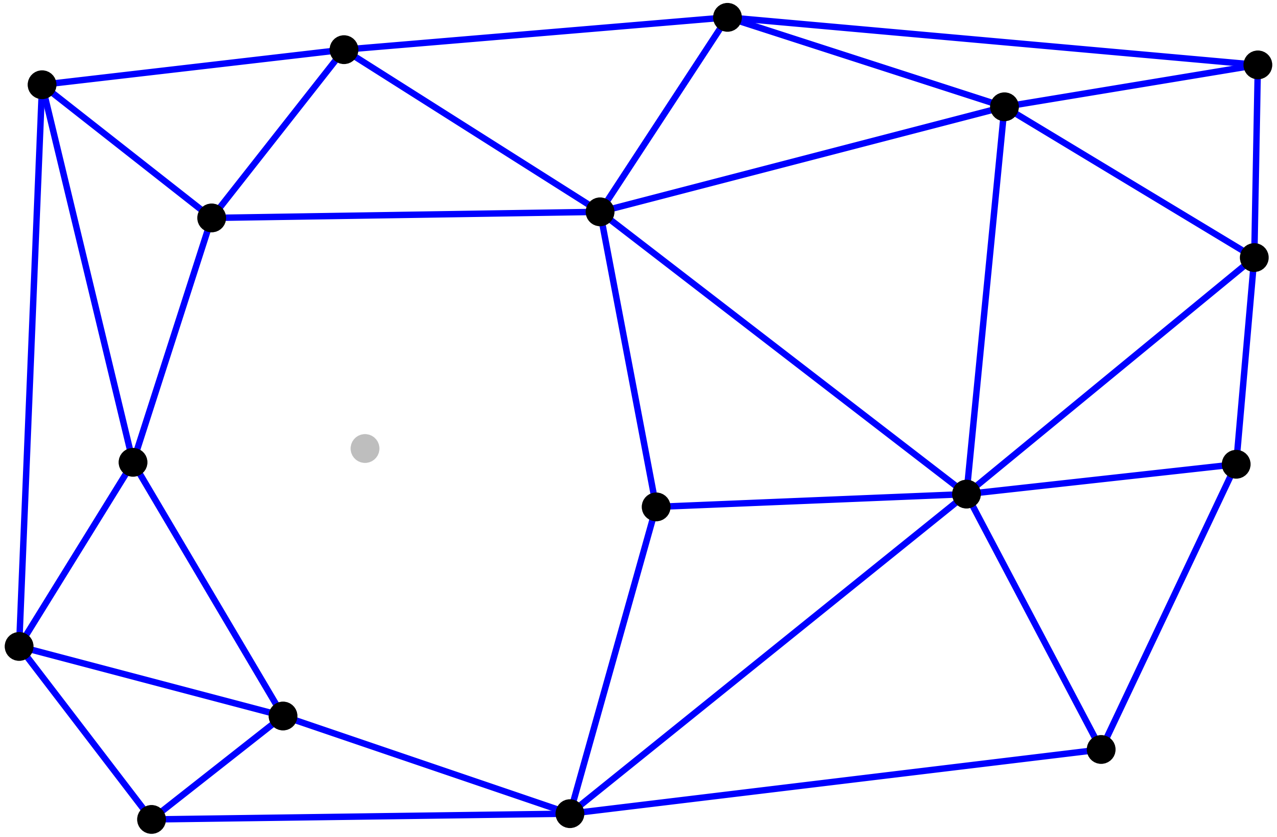


Delaunay Triangulation: deletion algorithm (sketch)



Delaunay Triangulation: deletion algorithm (sketch)

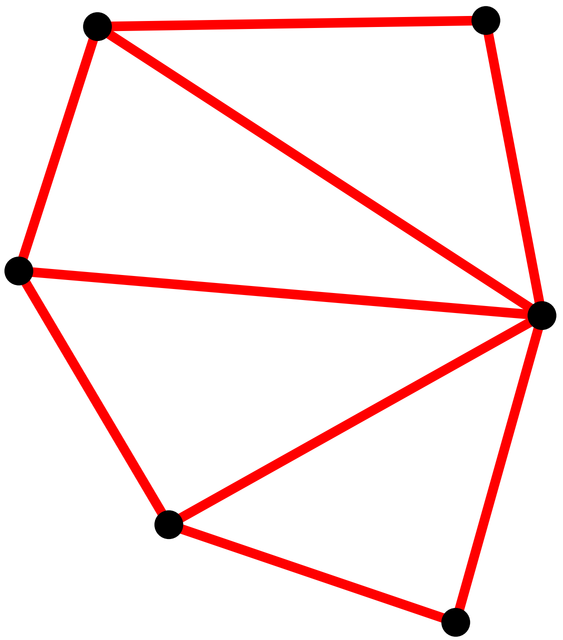
Extract hole

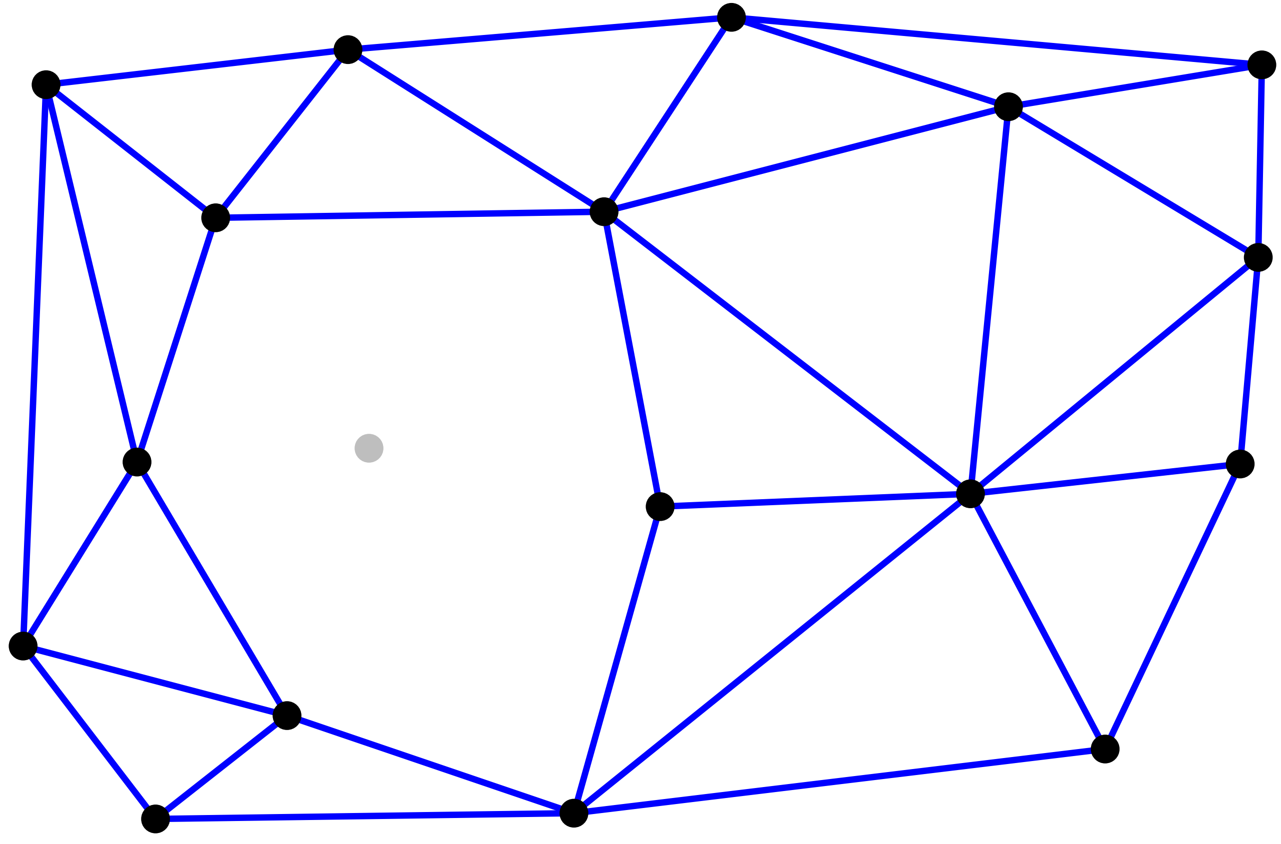


Delaunay Triangulation: deletion algorithm (sketch)

Extract hole

Triangulate

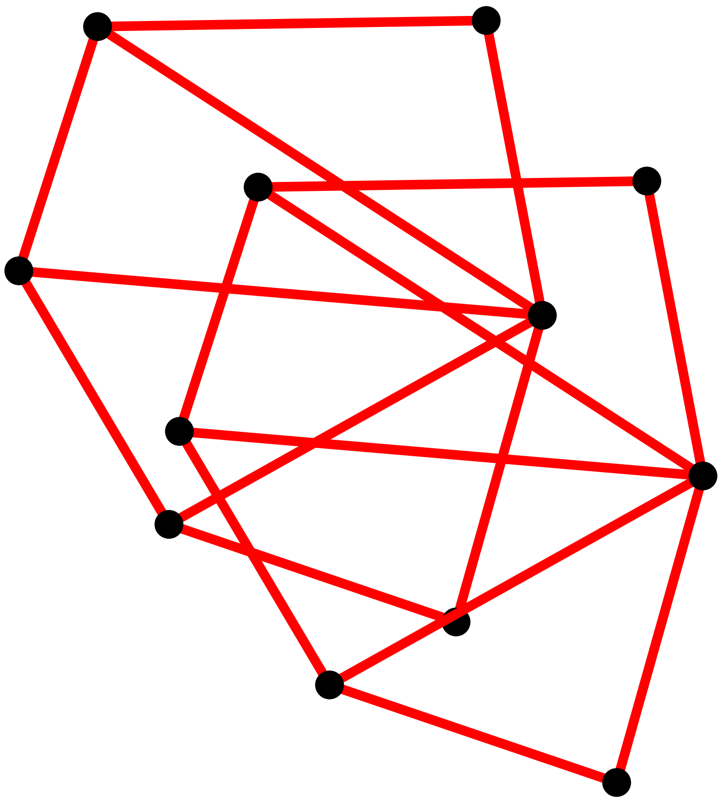


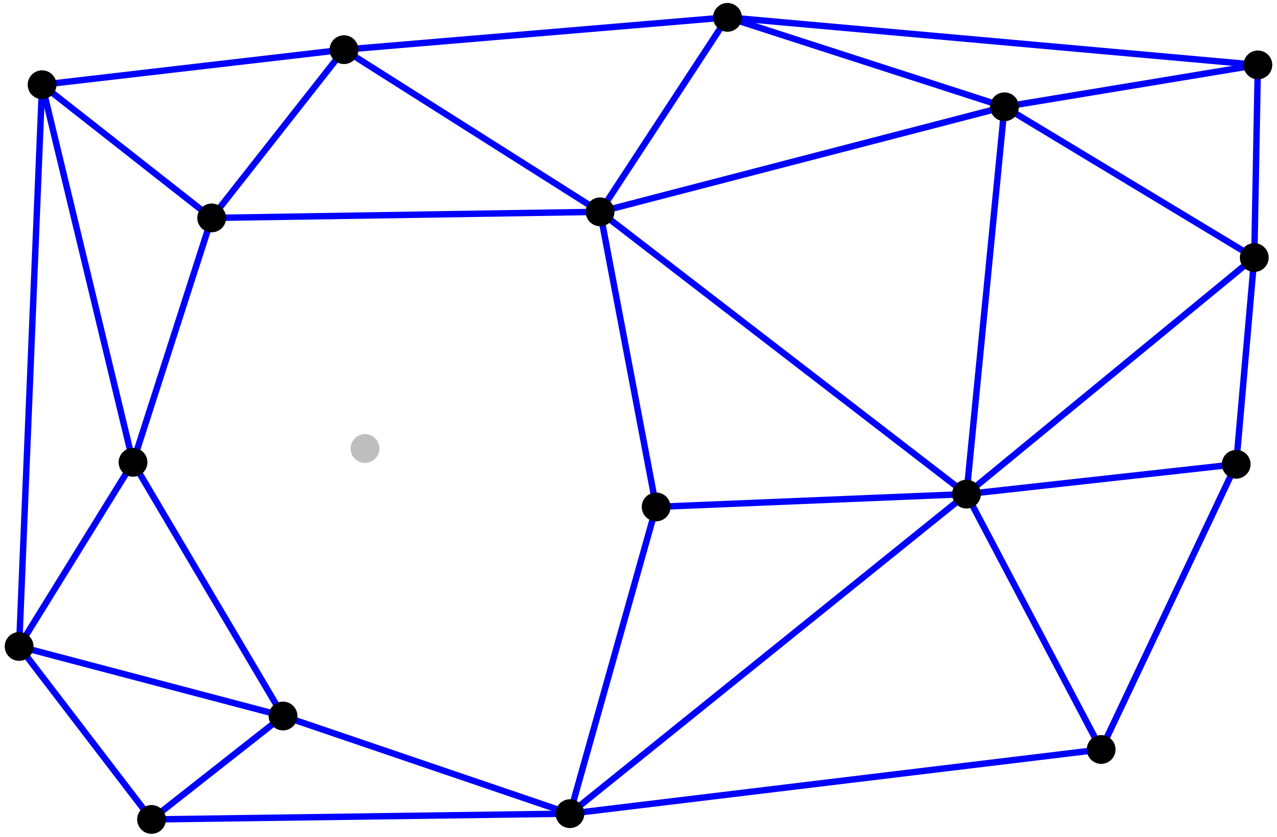


Delaunay Triangulation: deletion algorithm (sketch)

Extract hole

Triangulate

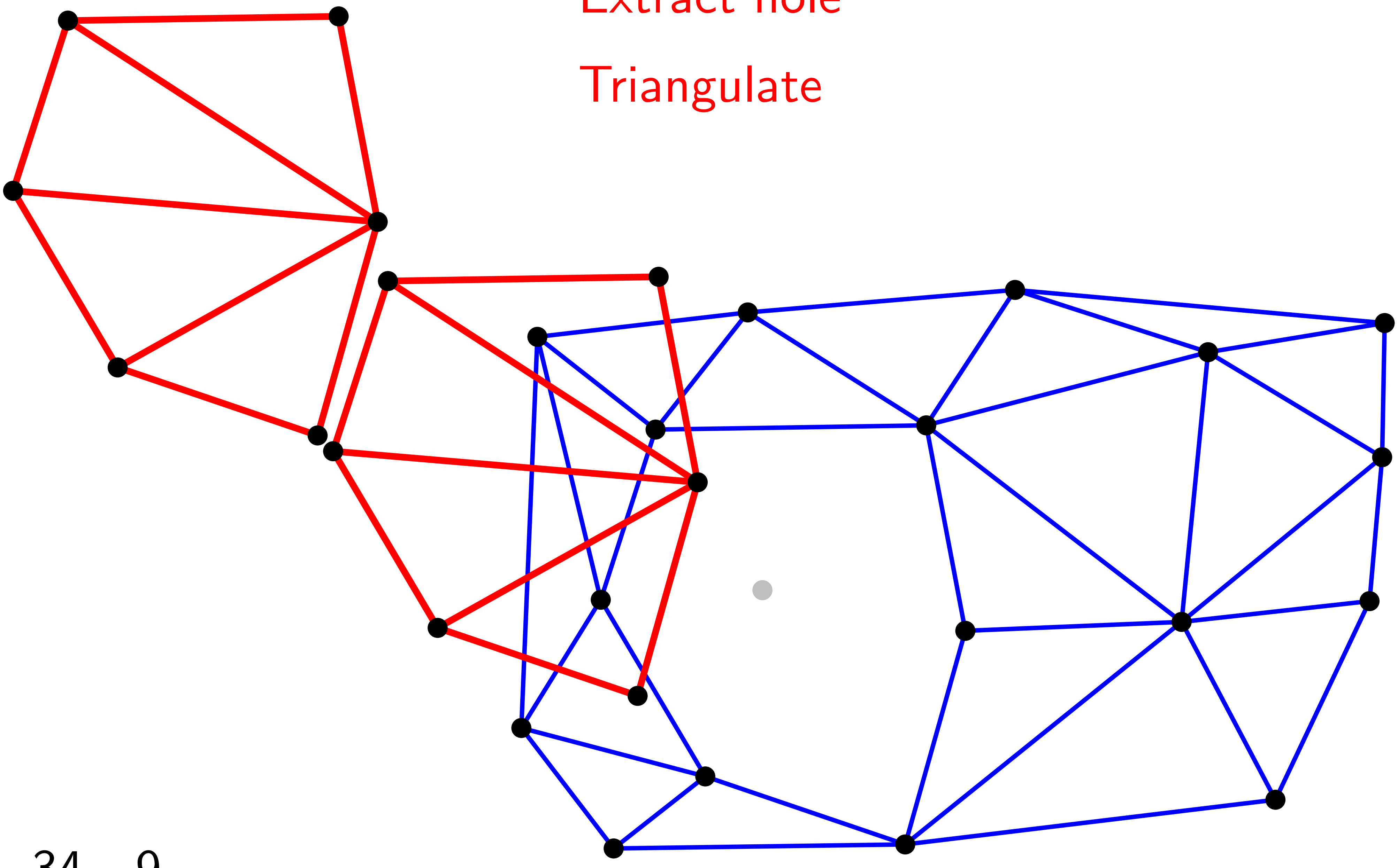




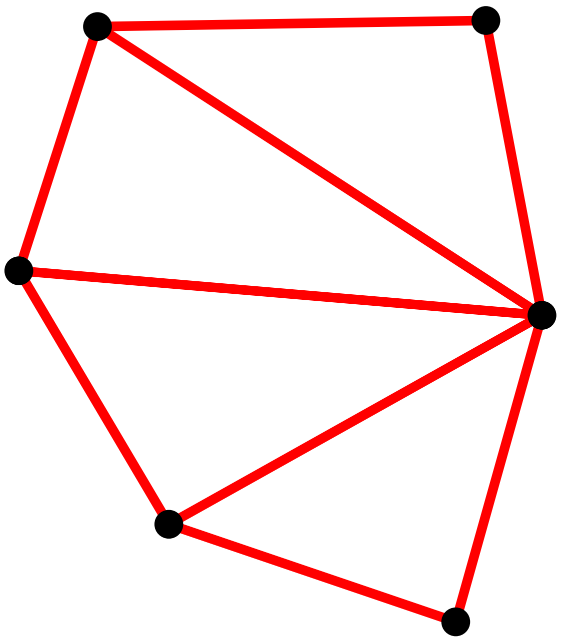
Delaunay Triangulation: deletion algorithm (sketch)

Extract hole

Triangulate



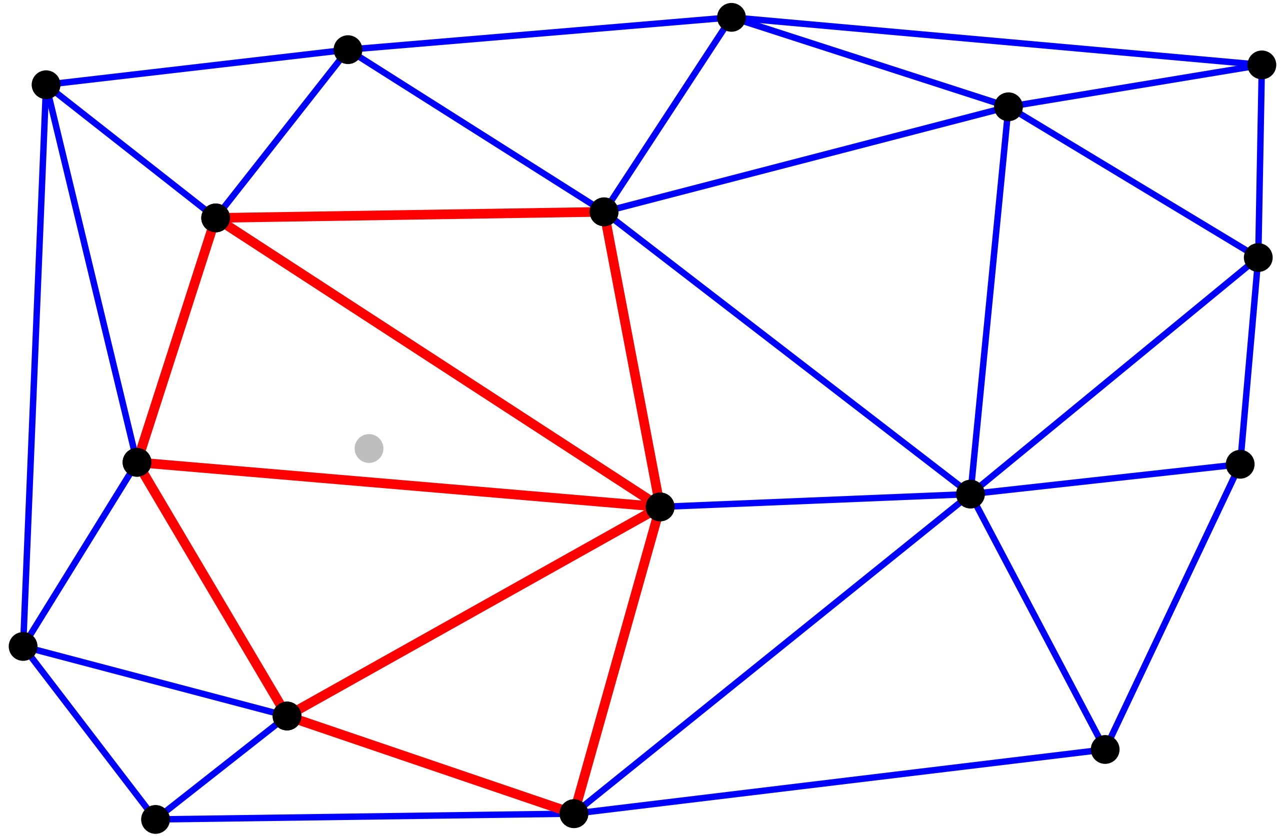
Delaunay Triangulation: deletion algorithm (sketch)



Extract hole

Triangulate

and sew

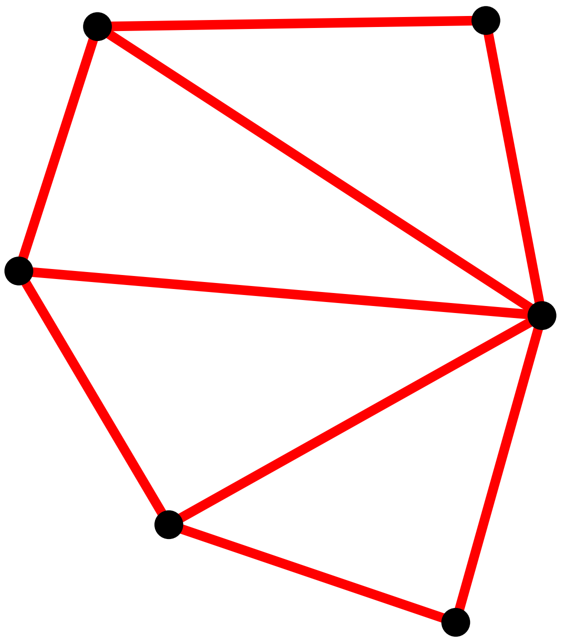


Delaunay Triangulation: deletion algorithm (sketch)

Extract hole

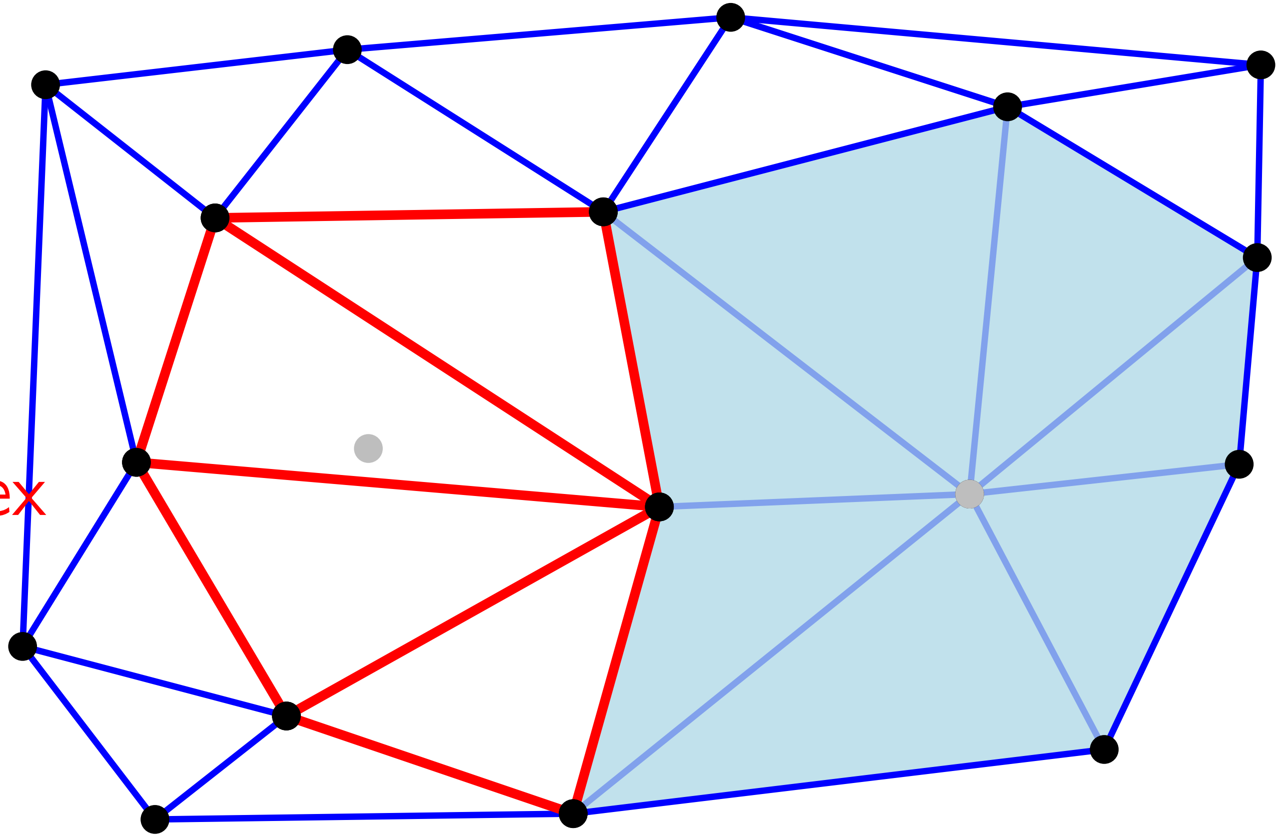
Triangulate

and sew



Be careful

Hole may be not convex

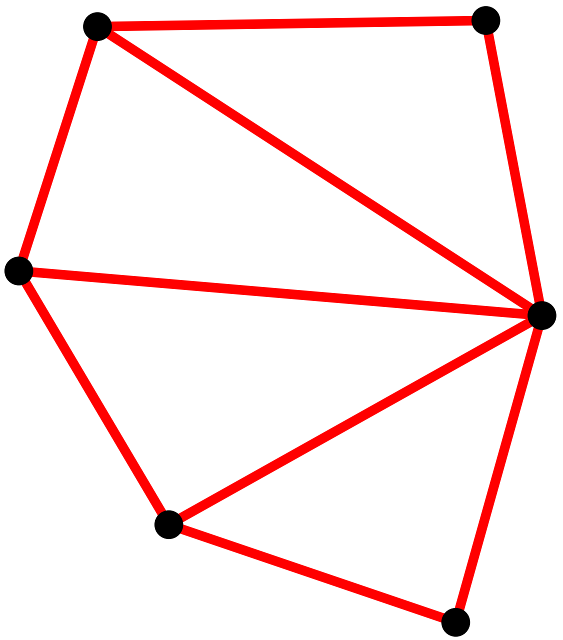


Delaunay Triangulation: deletion algorithm (sketch)

Extract hole

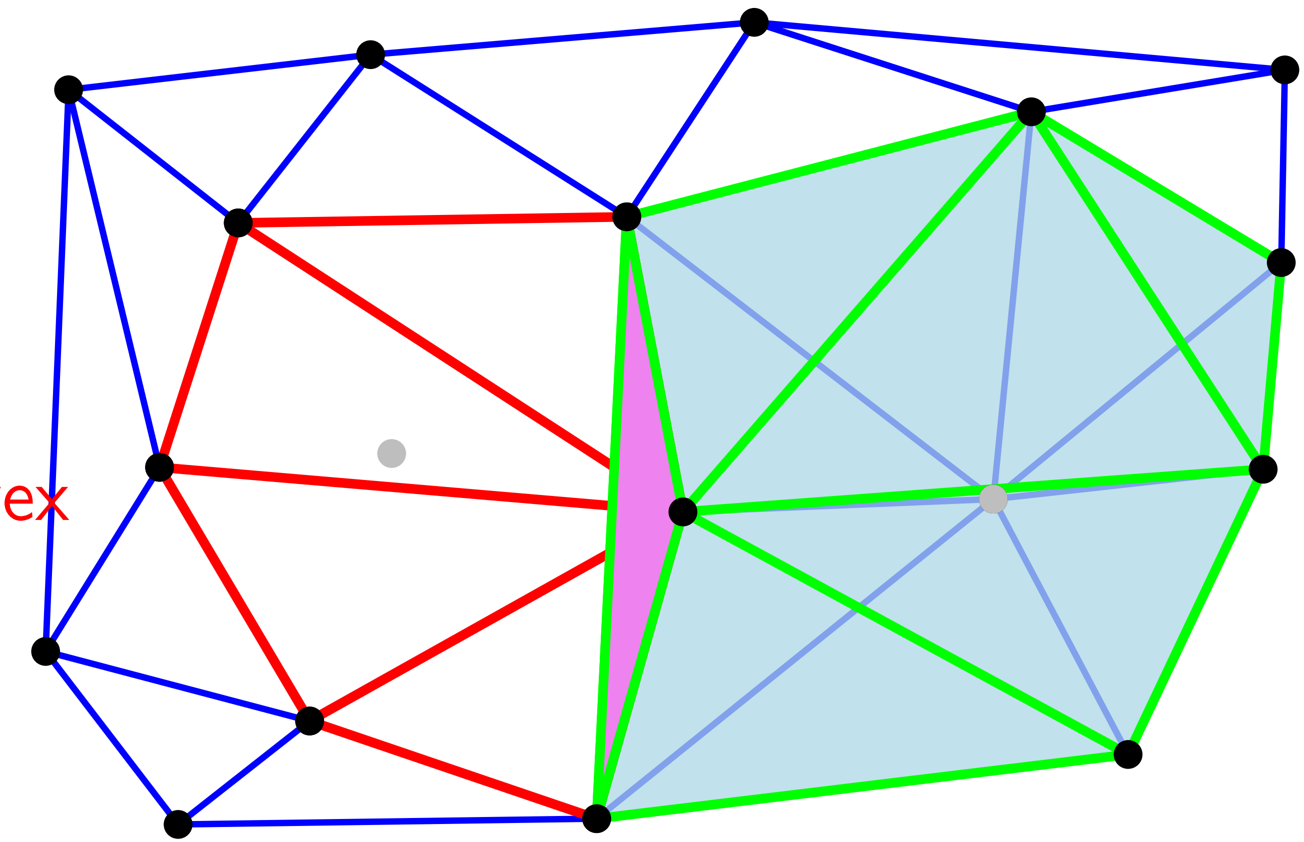
Triangulate

and sew



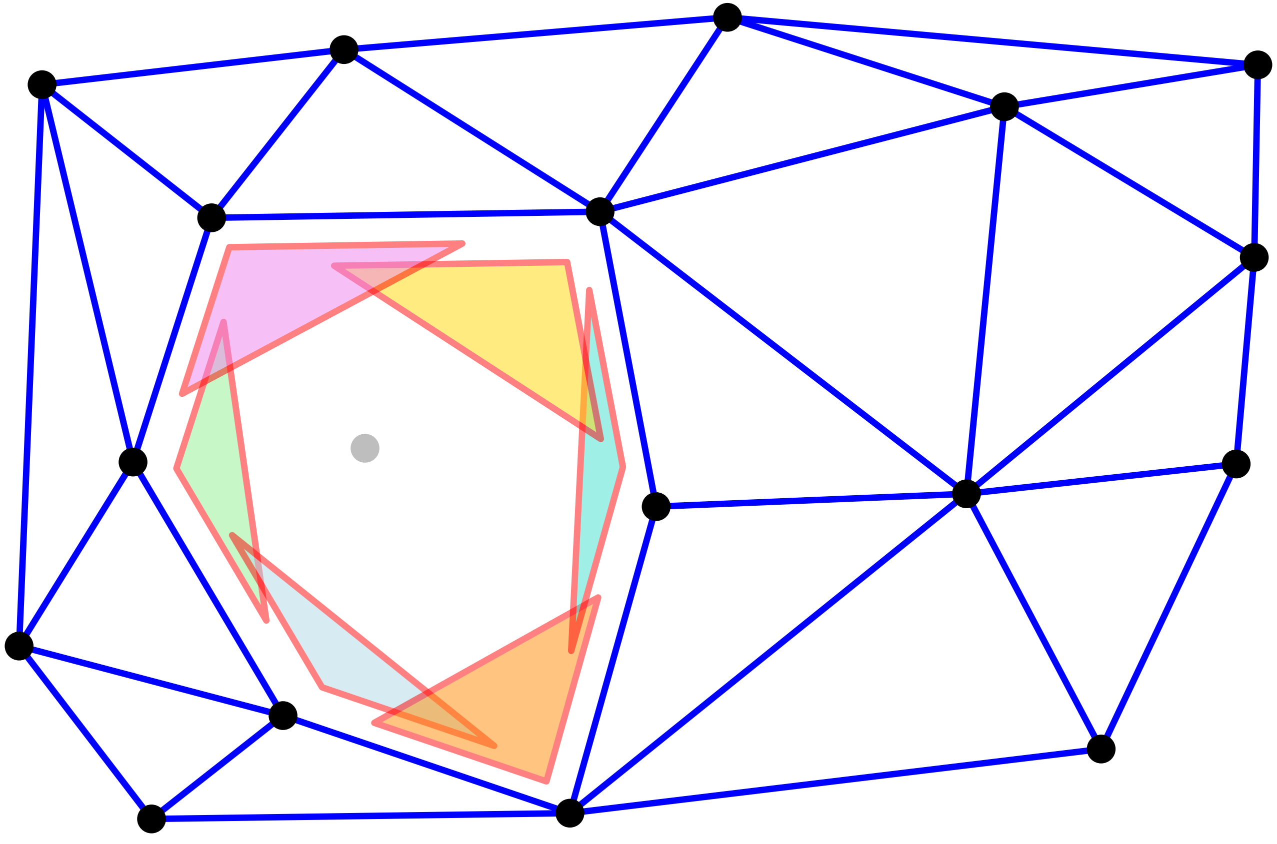
Be careful

Hole may be not convex



Delaunay Triangulation: deletion algorithm (sketch)

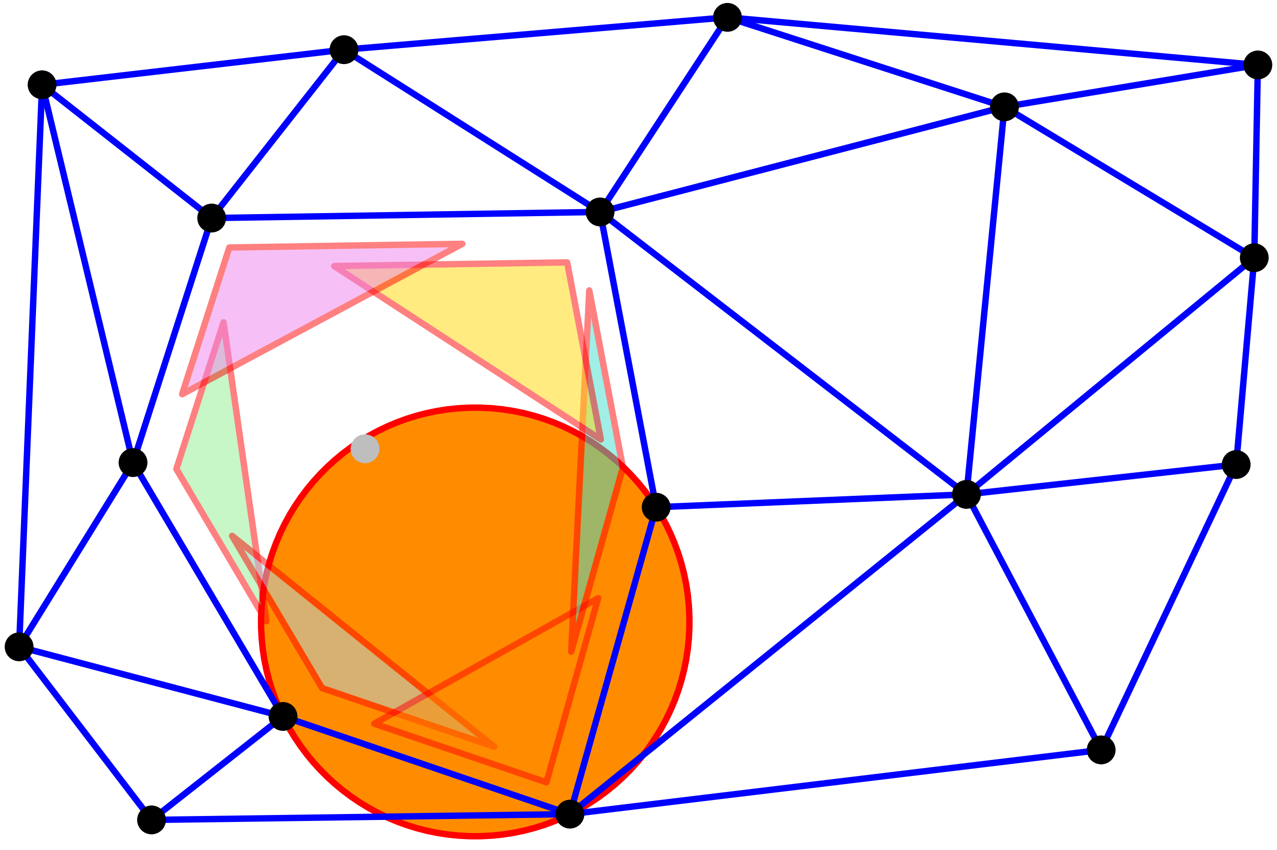
Ear queue



Delaunay Triangulation: deletion algorithm (sketch)

Ear queue

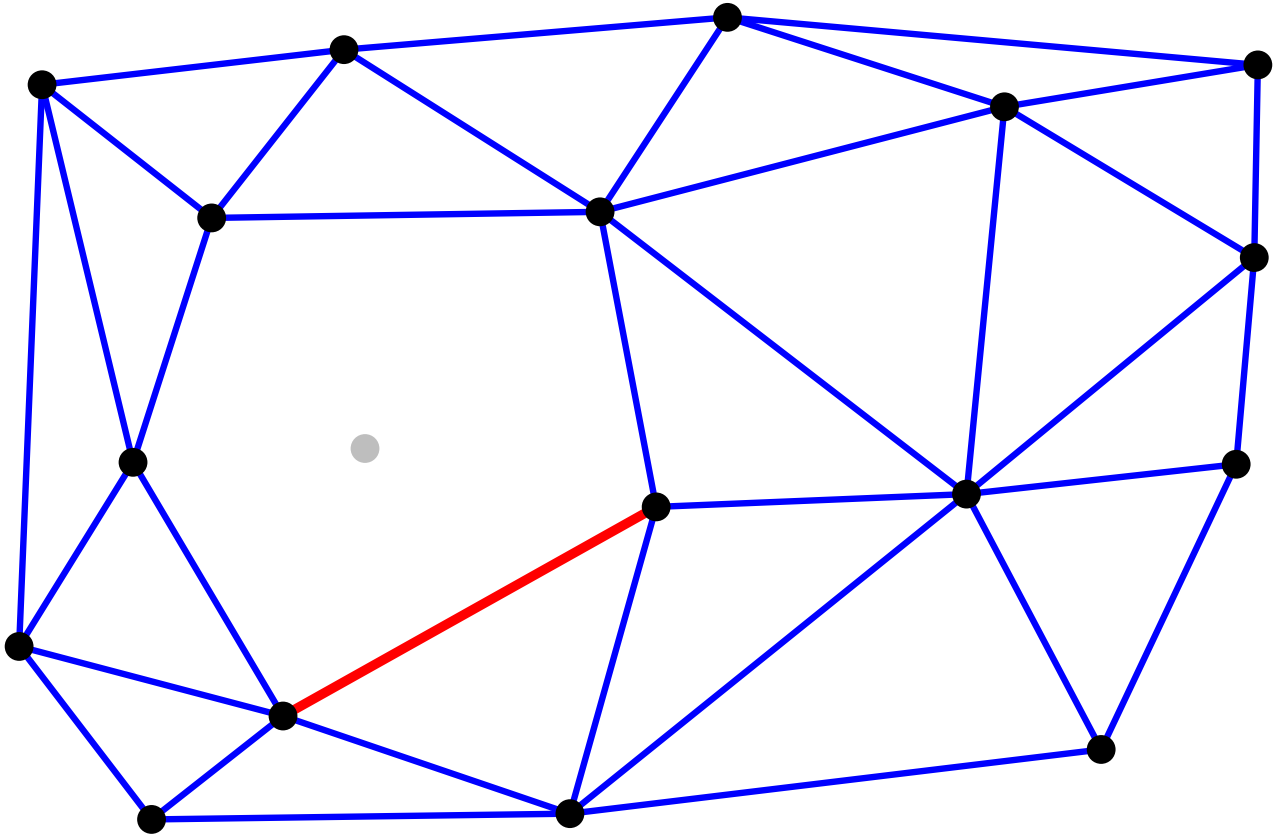
Ear with largest power is added



Delaunay Triangulation: deletion algorithm (sketch)

Ear queue

Ear with largest power is added

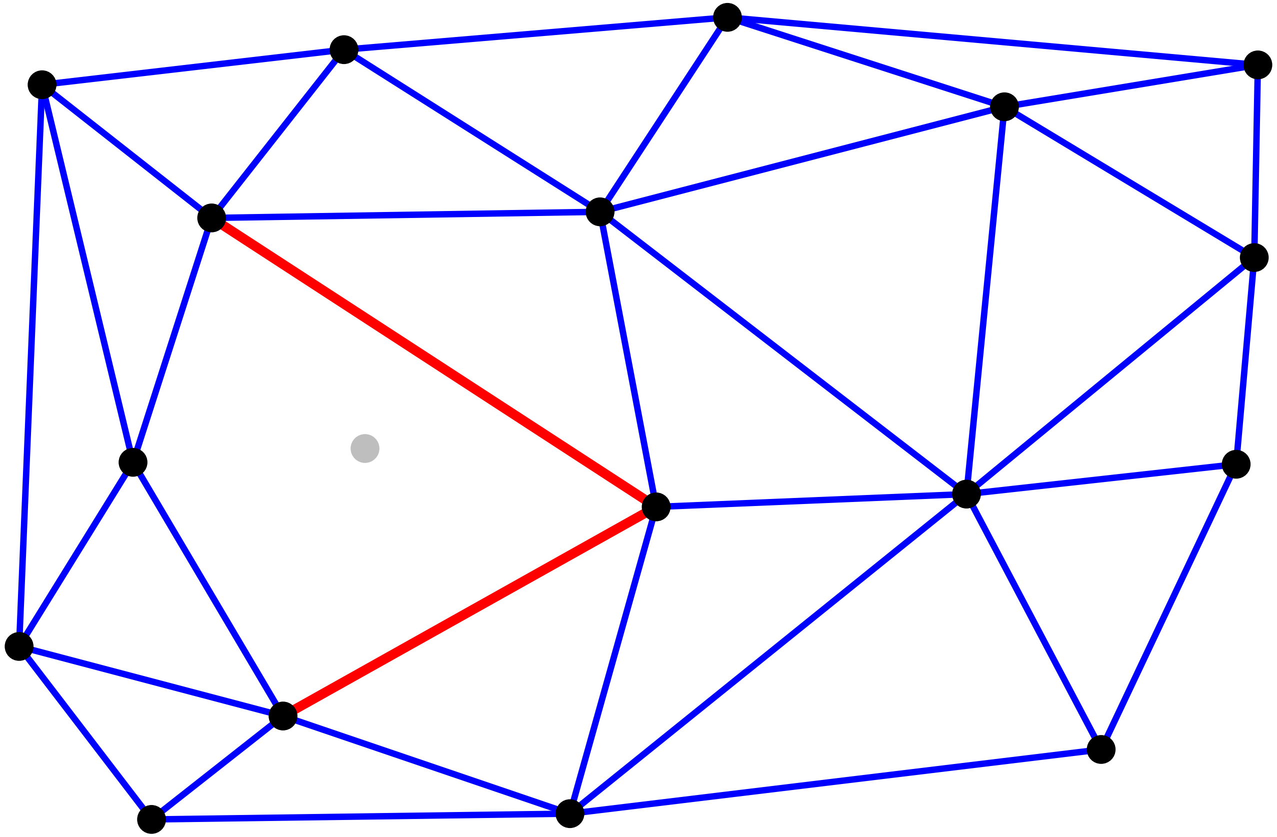


Delaunay Triangulation: deletion algorithm (sketch)

Ear queue

Ear with largest power is added

Iterate

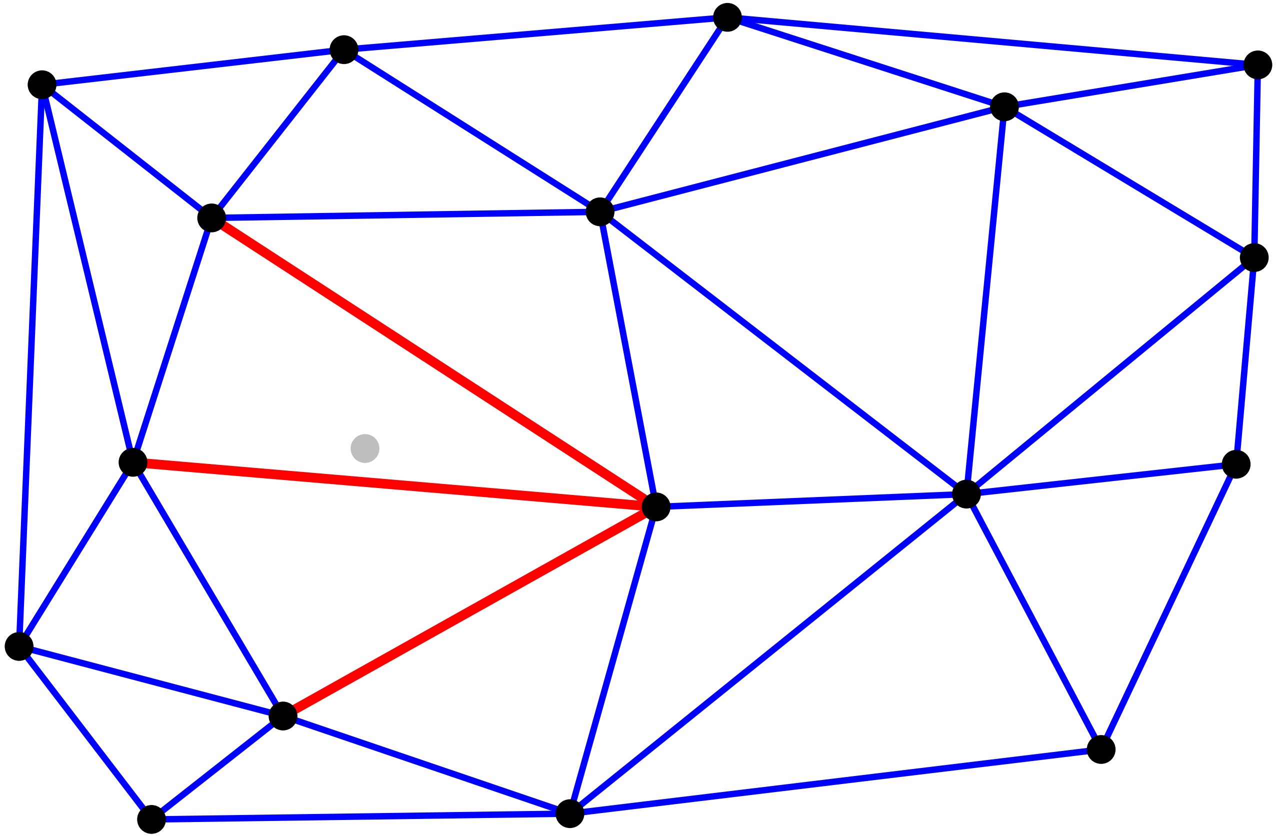


Delaunay Triangulation: deletion algorithm (sketch)

Ear queue

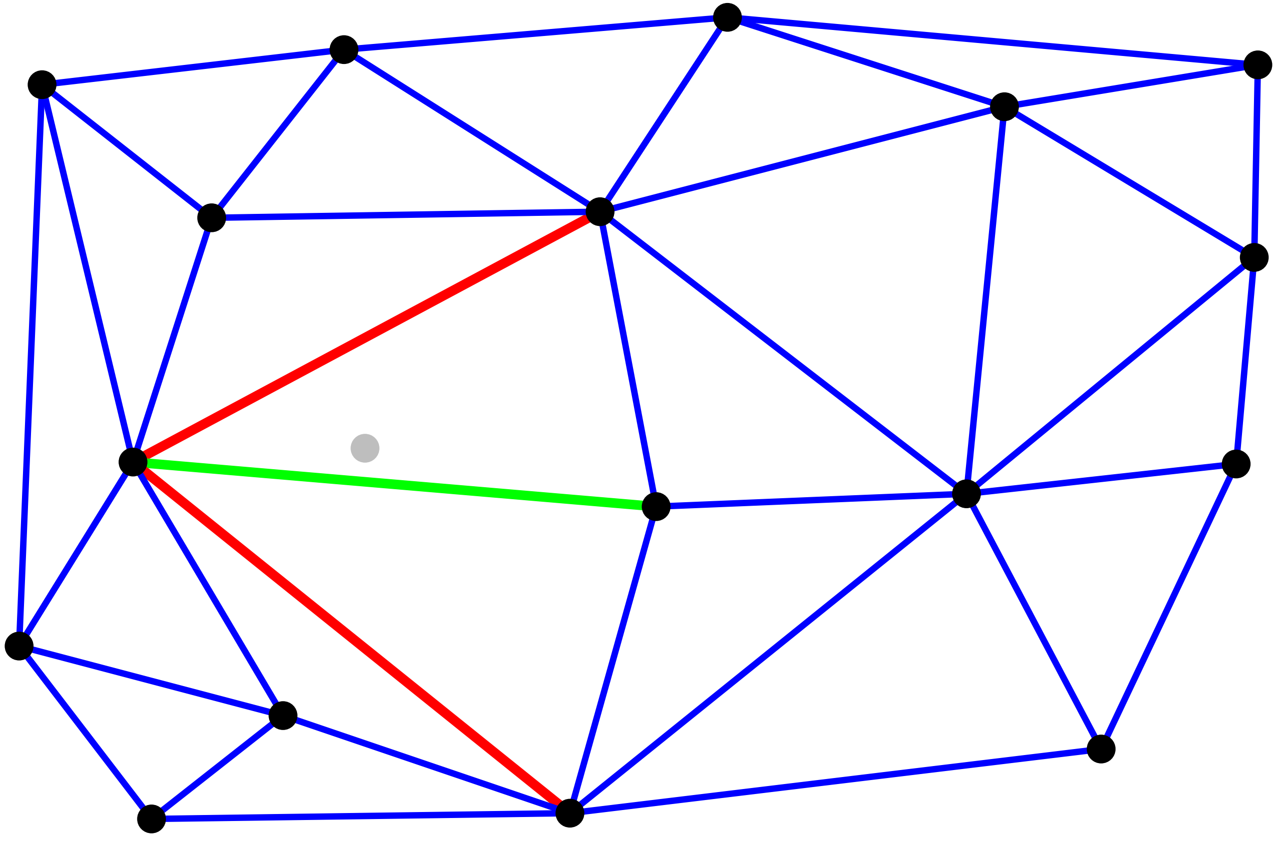
Ear with largest power is added

Iterate



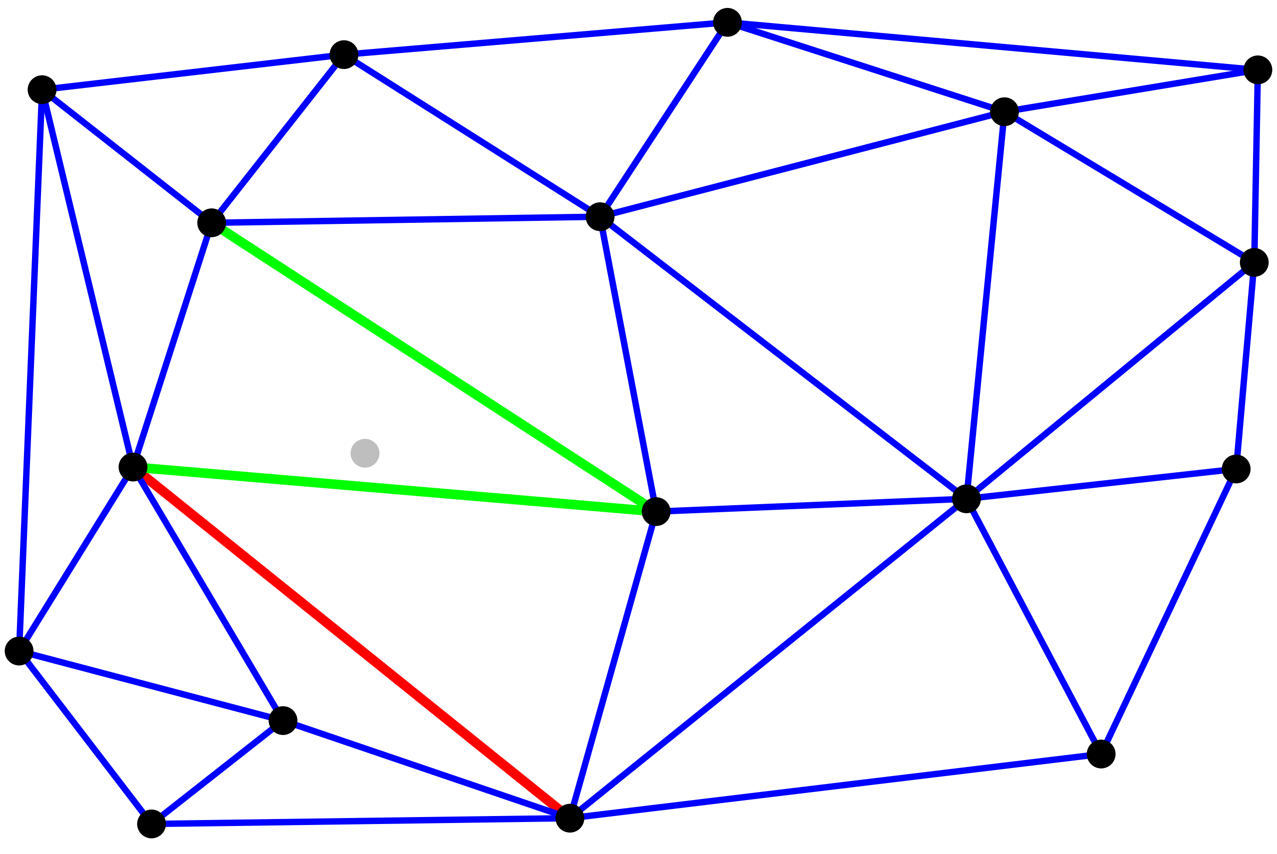
Delaunay Triangulation: deletion algorithm (sketch)

Triangulate and flip



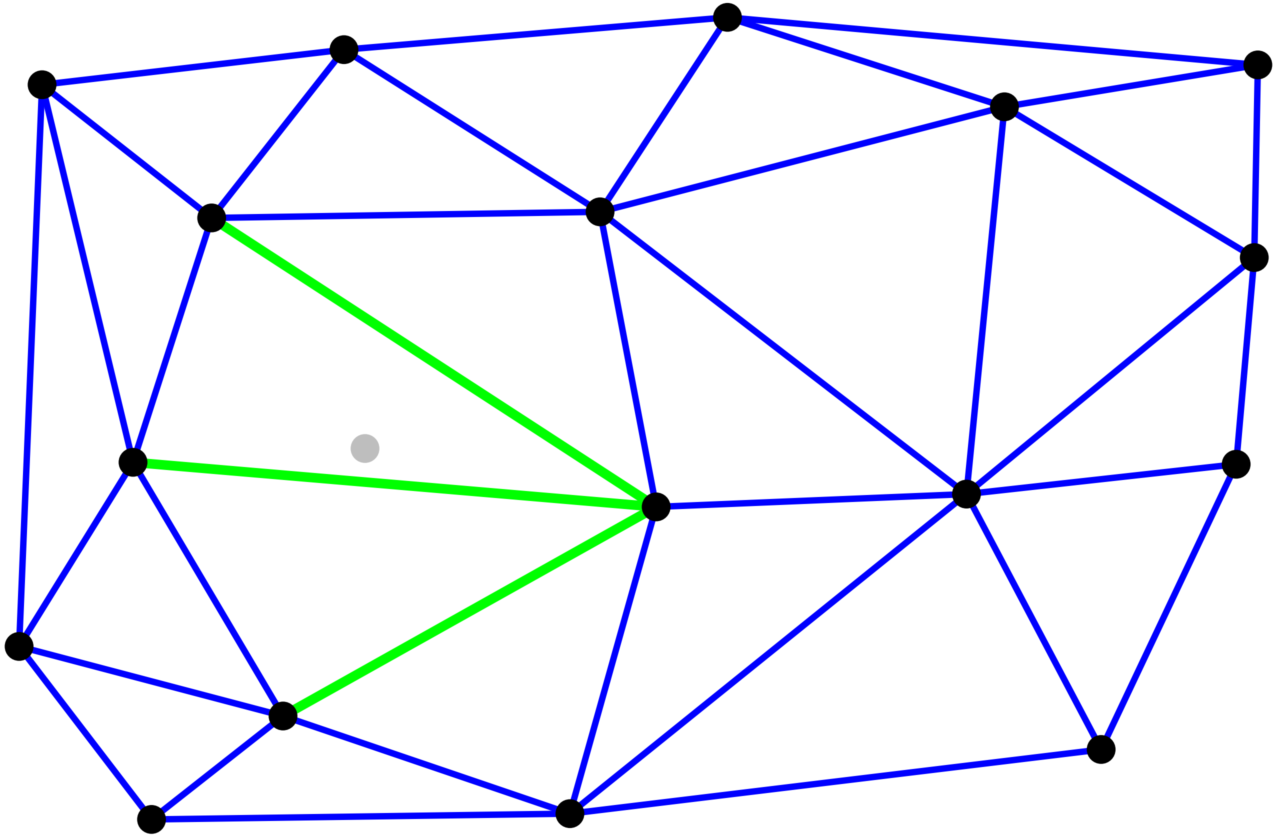
Delaunay Triangulation: deletion algorithm (sketch)

Triangulate and flip



Delaunay Triangulation: deletion algorithm (sketch)

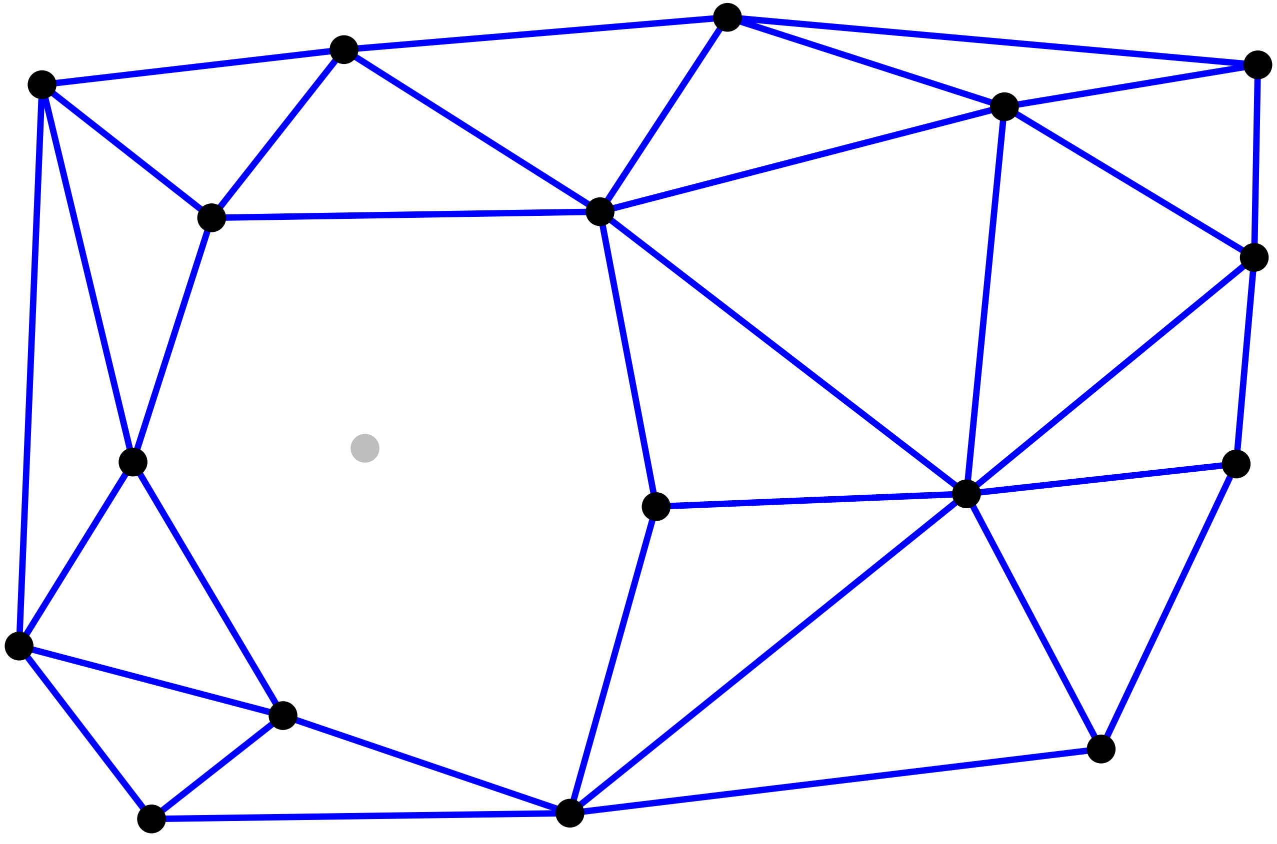
Triangulate and flip



for degree ≥ 8

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes



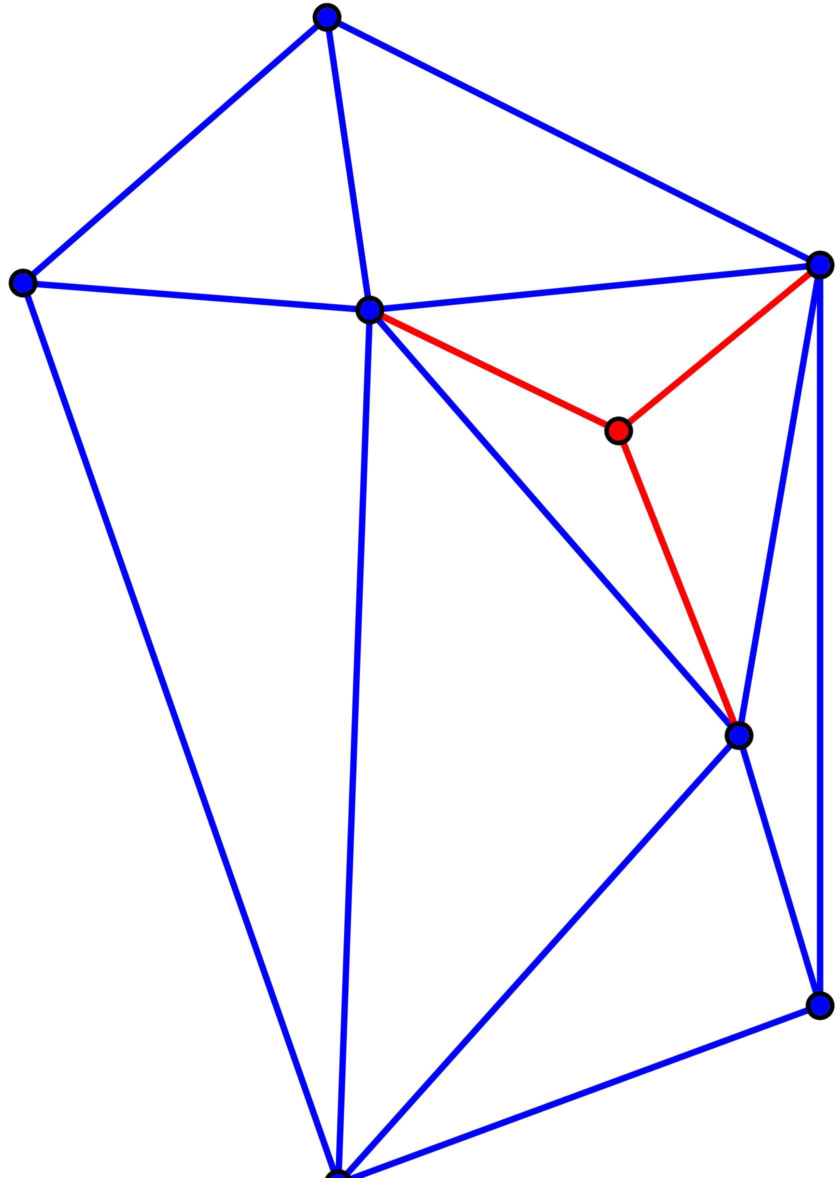
for degree ≤ 7

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes

degree 3

nothing to do



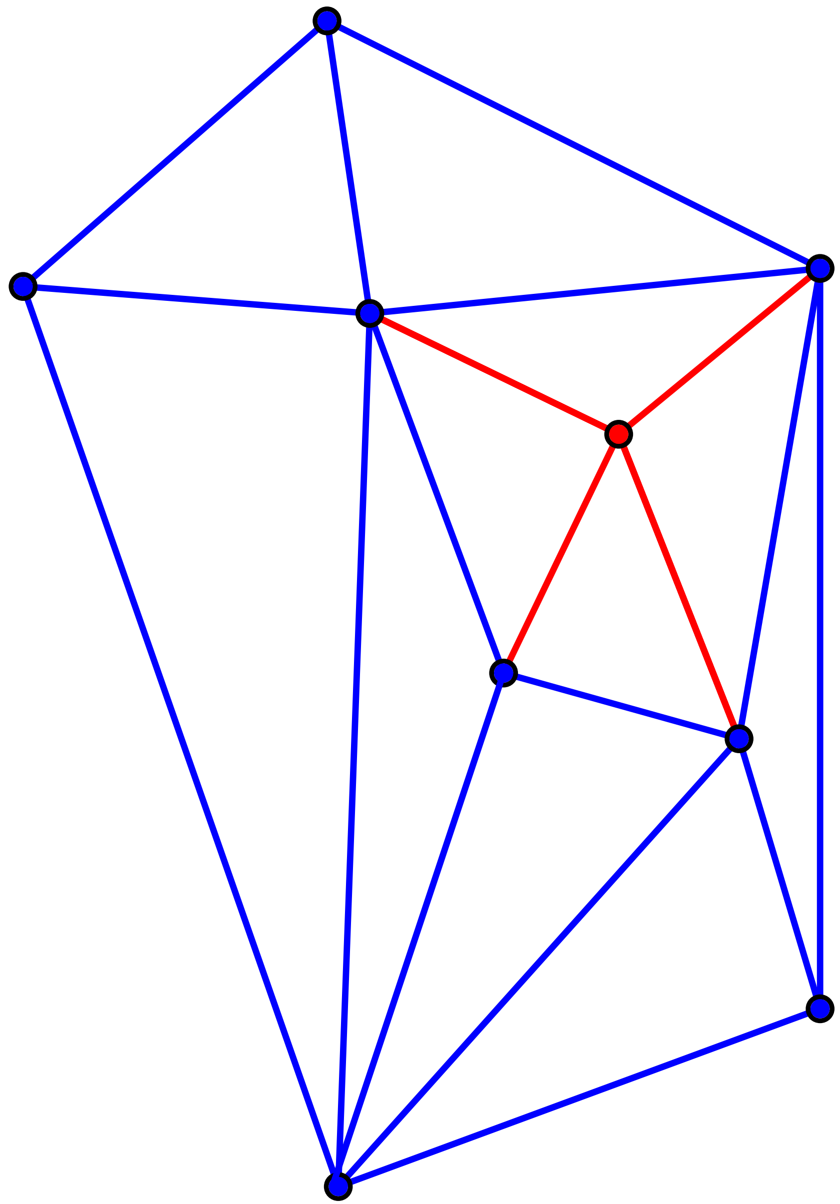
CGAL

for degree ≤ 7

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes

degree 4



CGAL

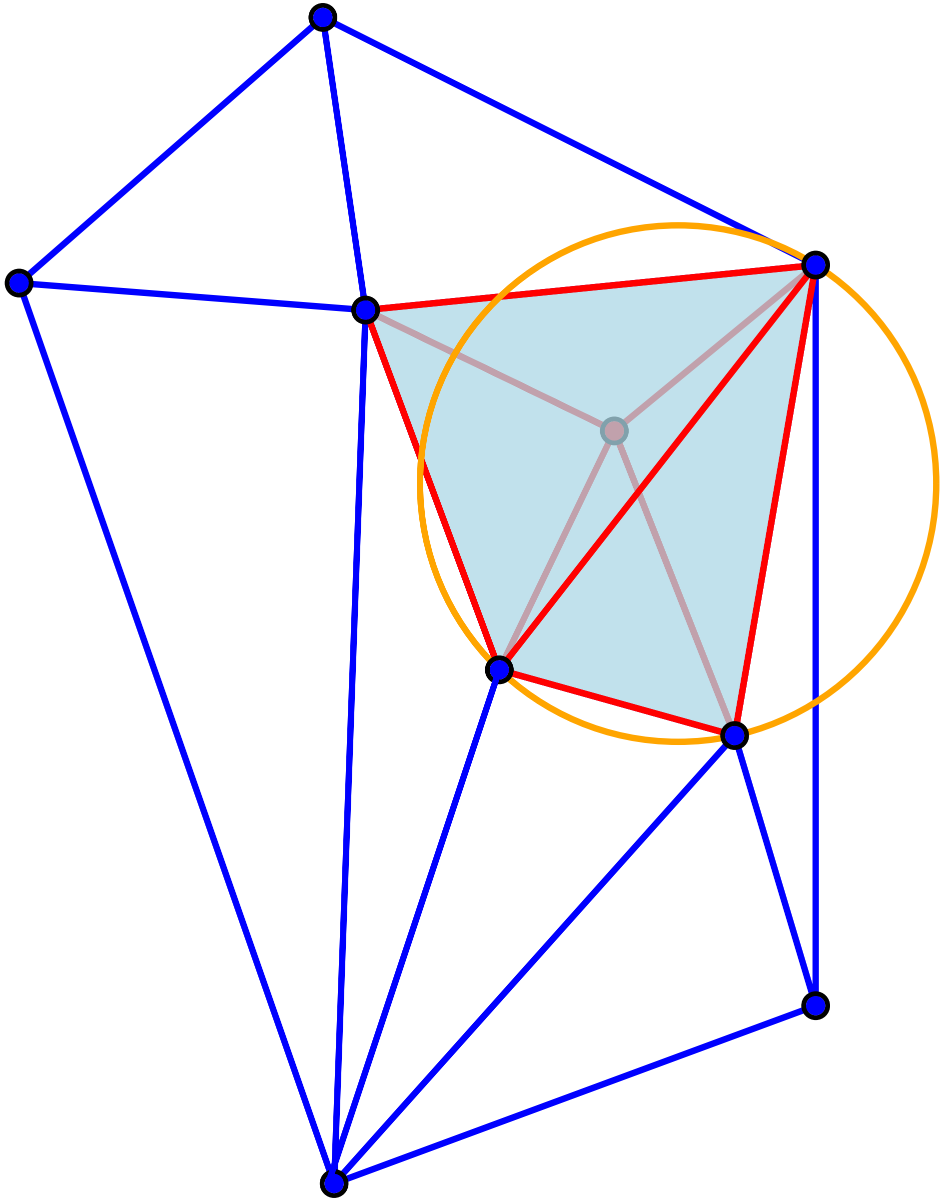
for degree ≤ 7

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes

degree 4

only one predicate



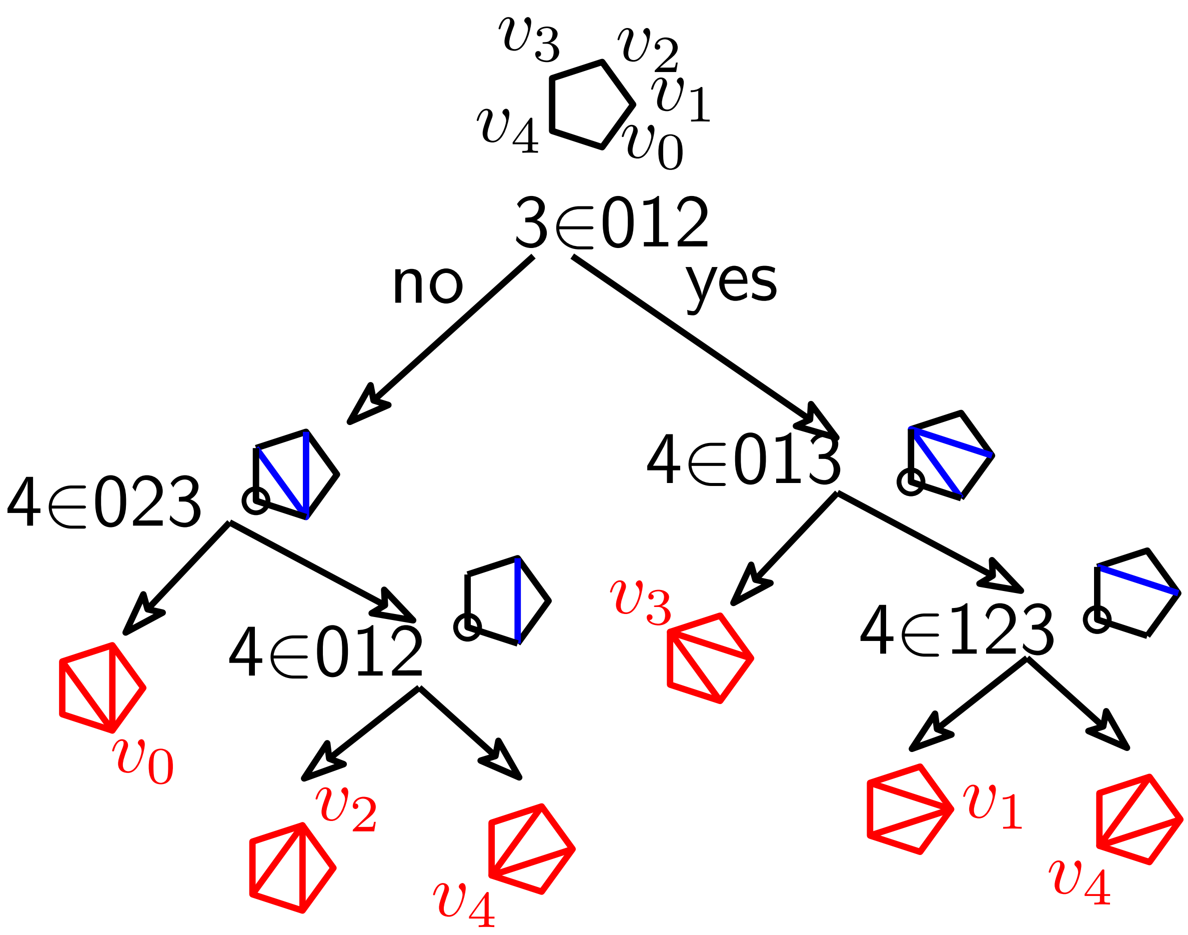
CGAL

for degree ≤ 7

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes

degree 5



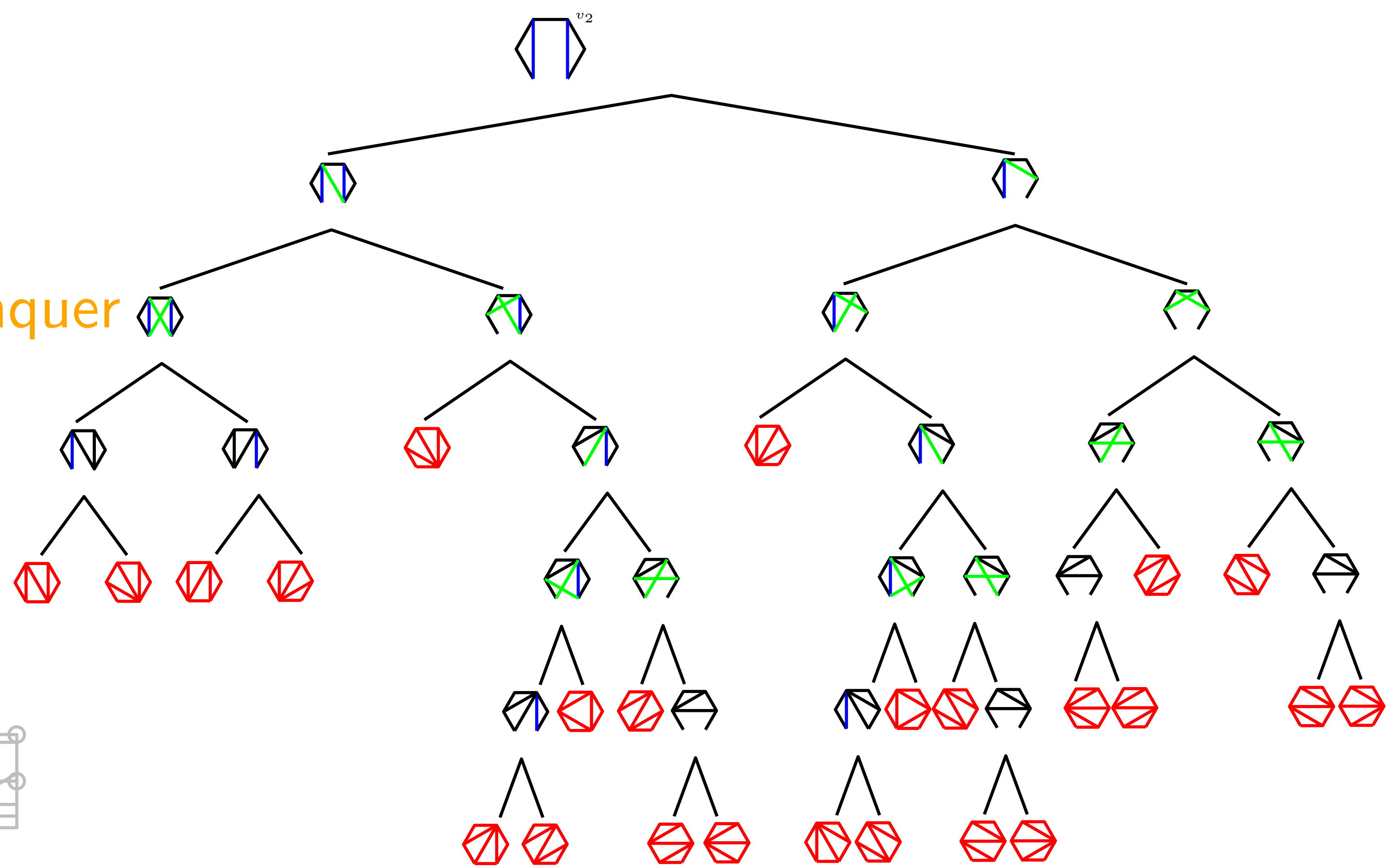
for degree ≤ 7

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes

degree 6

"manual" d&conquer



CGAL

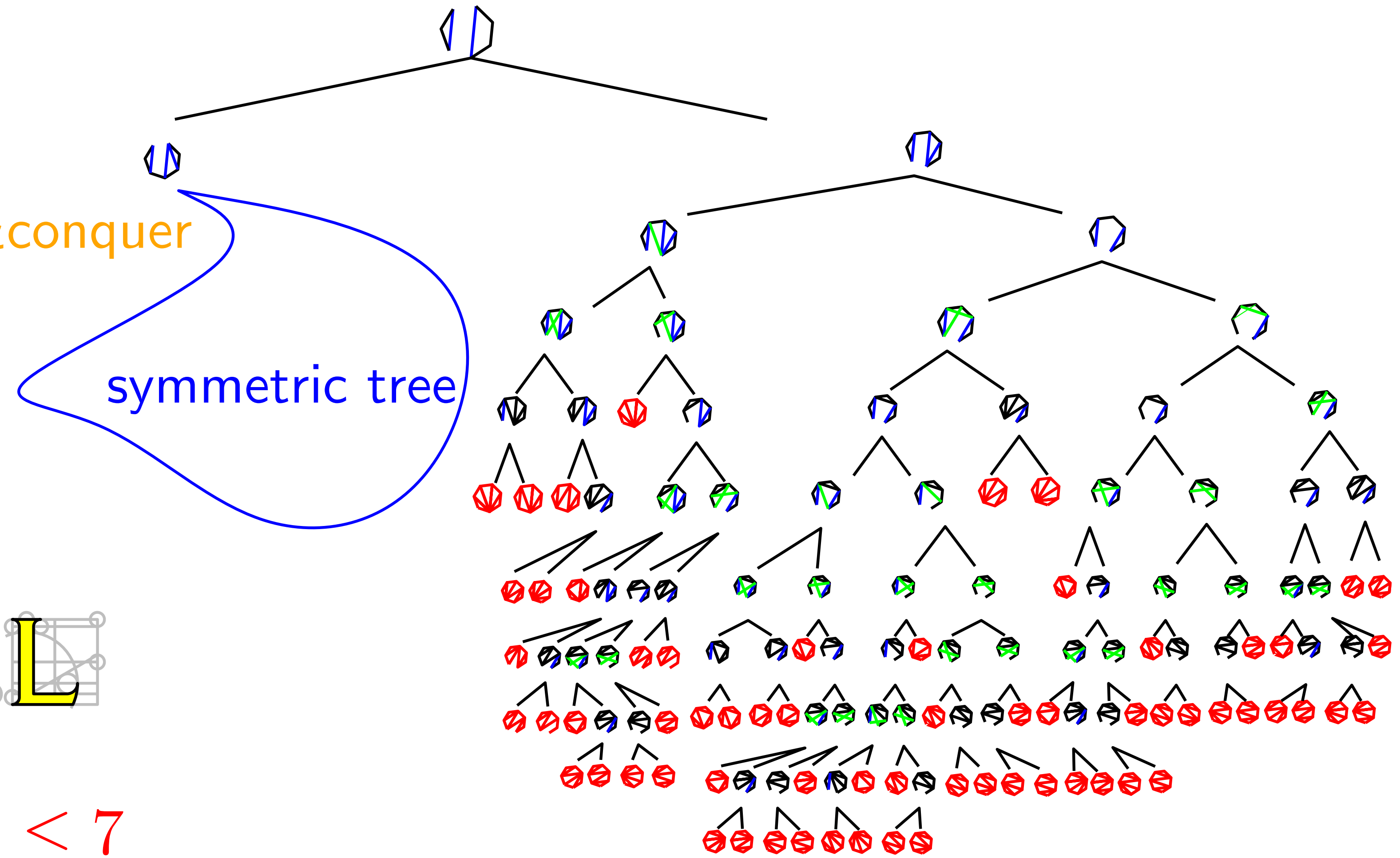
for degree ≤ 7

Delaunay Triangulation: deletion algorithm (sketch)

Decision tree for small holes

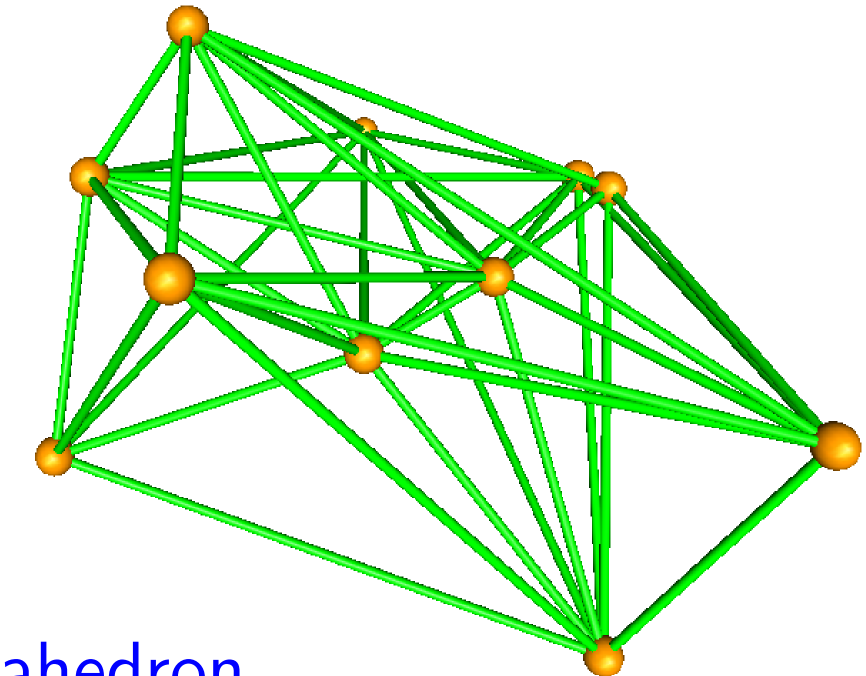
degree 7

"manual" d&conquer



for degree ≤ 7

Delaunay Triangulation: 3D



Same as 2D

Dual Voronoi diagram

Empty sphere property

Triangle $\longrightarrow$ Tetrahedron

Duality with 4D convex hull

Incremental algorithm (find the hole and star)

Delaunay T

Same as 2D

Dual

Empty

Tri

Convex hull

Higher dimensions

Dehn Sommerville relations $\quad f_i = \sharp(\text{faces of dim } i)$

Euler: $\quad f_0 - f_1 + f_2 - \dots f_{d-1} = (-1)^{d-1} + 1$

$$\sum_j = k^{d-1} - 1^j \begin{pmatrix} j+1 \\ k+1 \end{pmatrix} f_j = (-1)^{d-1} f_k$$

$-1 \leq k \leq d-2 \qquad f_{-1} = f_d = 1$

$\left\lfloor \frac{d+1}{2} \right\rfloor$ independent equations

Duality with 4D convex hull

Incremental algorithm (find the hole and star)

Delaunay T

Convex hull

Higher dimensions

Dehn Sommerville relations $\quad f_i = \sharp(\text{faces of dim } i)$

Euler: $\quad f_0 - f_1 + f_2 - \dots f_{d-1} = (-1)^{d-1} + 1$

$$\sum_j = k^{d-1} - 1^j \begin{pmatrix} j+1 \\ k+1 \end{pmatrix} f_j = (-1)^{d-1} f_k$$

$$-1 \leq k \leq d-2 \qquad\qquad f_{-1} = f_d = 1$$

$\left\lfloor \frac{d+1}{2} \right\rfloor$ independent equations

Same as 2D

Dual

Empty

Tri

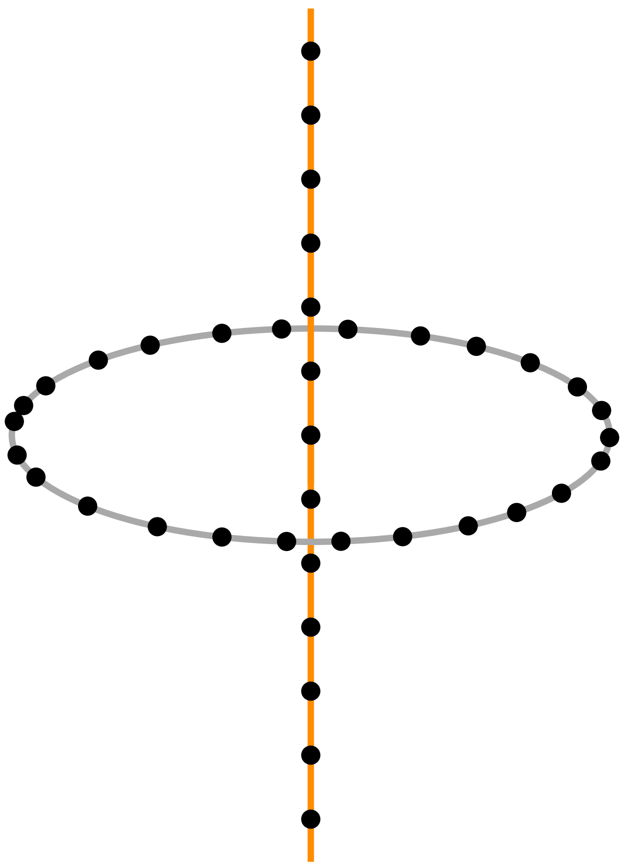
Duality with 4D convex hull

quadratic

Incremental algorithm (find the hole and star)

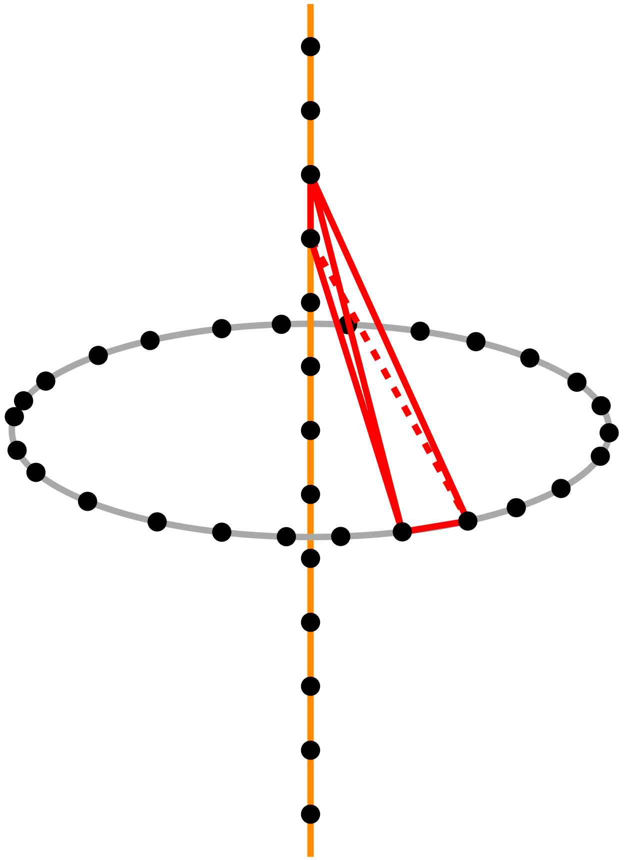
Delaunay Triangulation: 3D

Quadratic examples



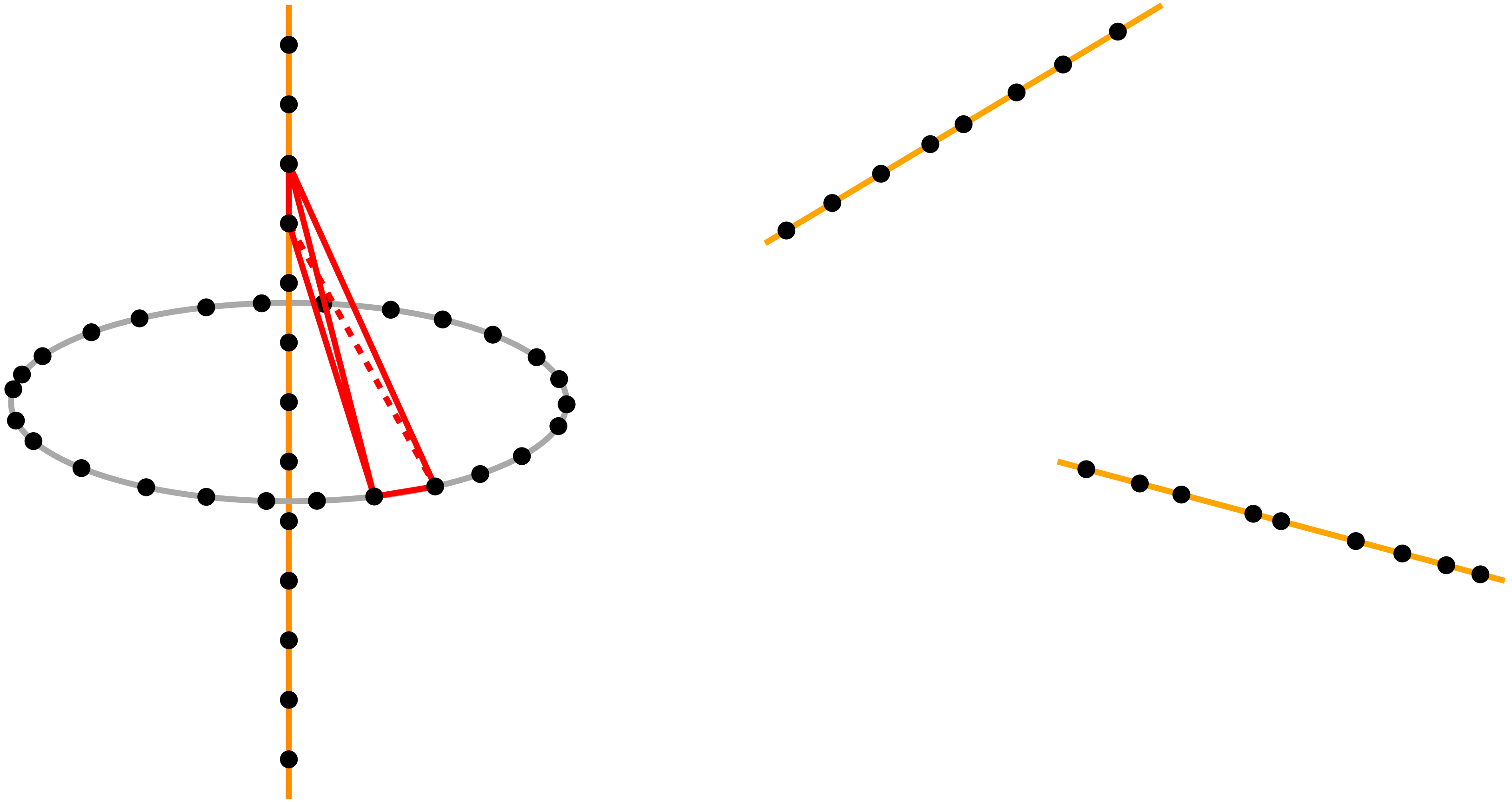
Delaunay Triangulation: 3D

Quadratic examples



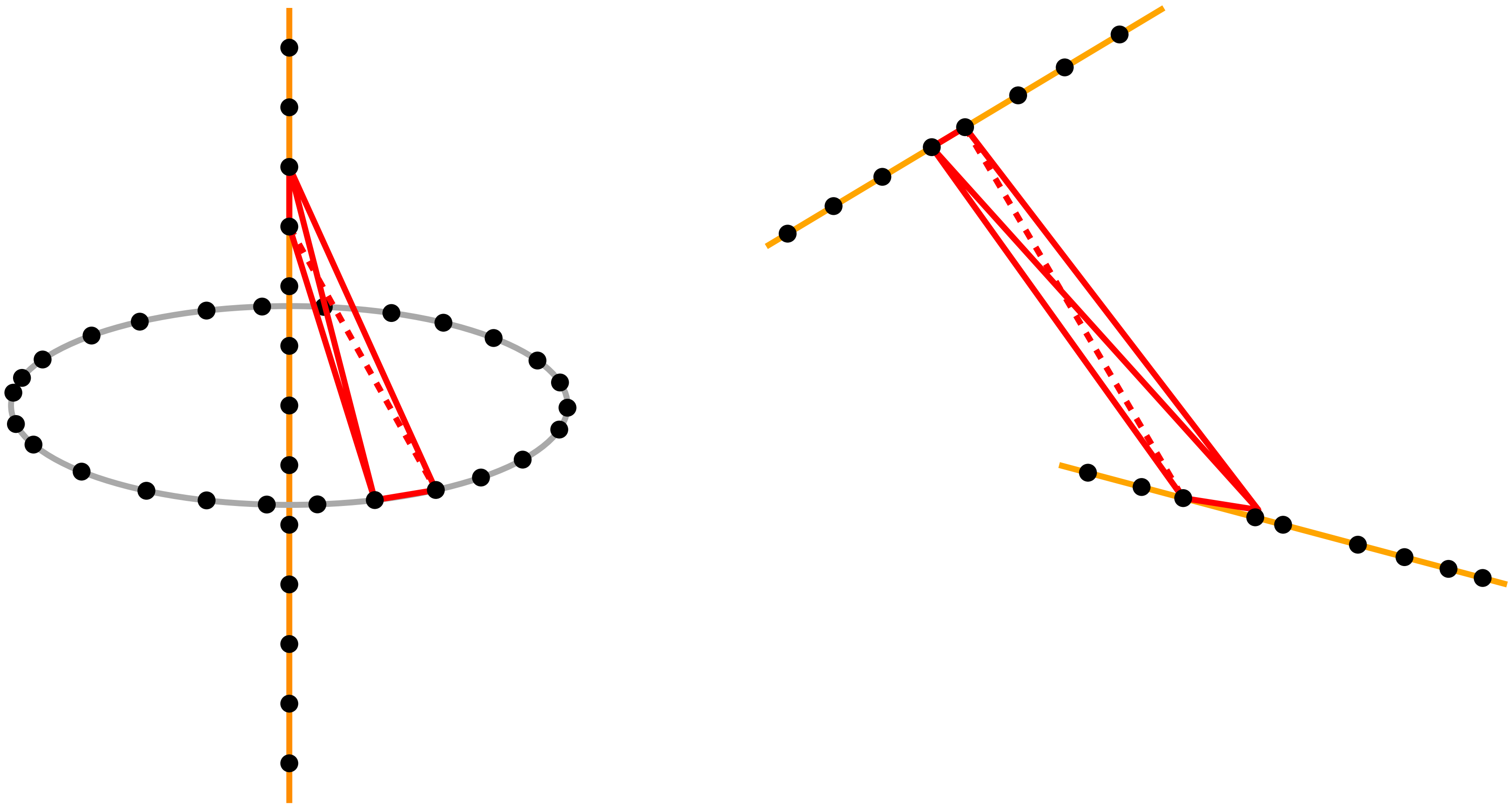
Delaunay Triangulation: 3D

Quadratic examples



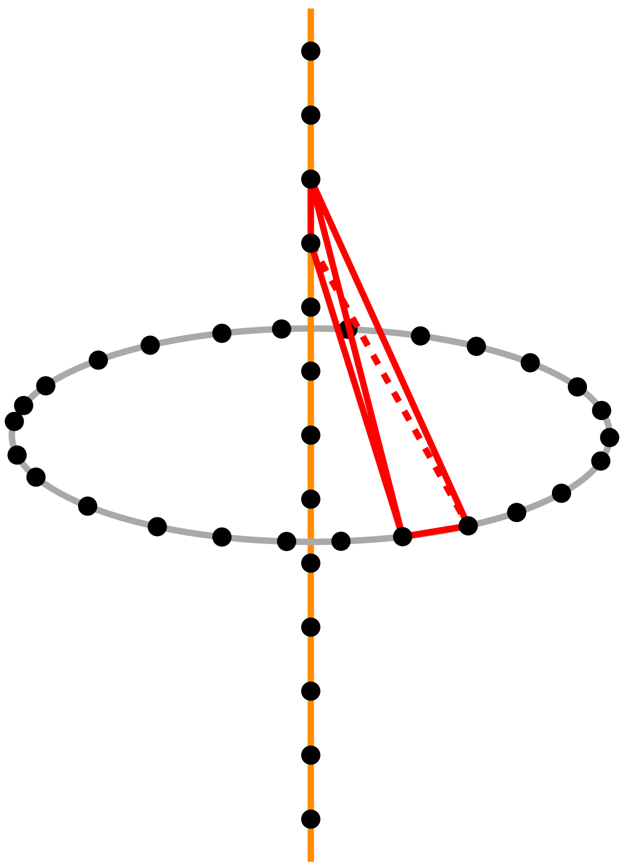
Delaunay Triangulation: 3D

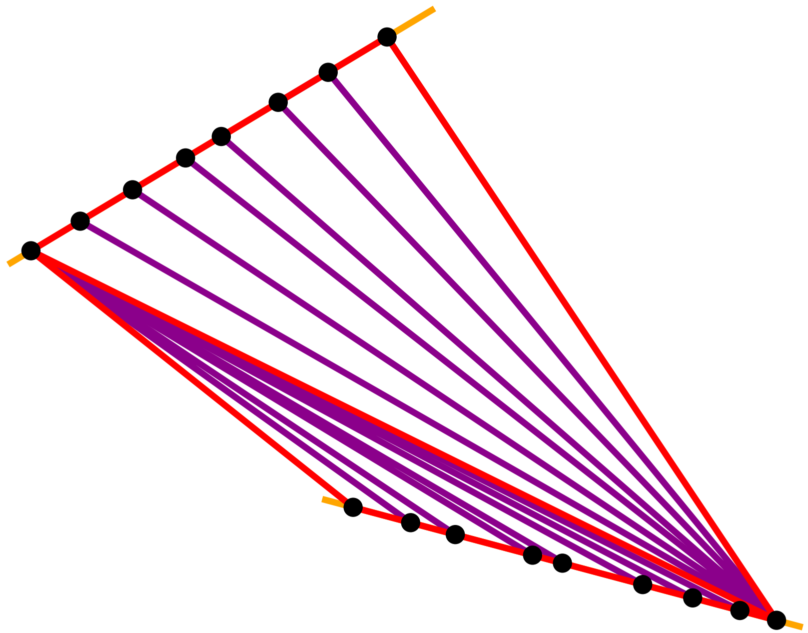
Quadratic examples



Delaunay Triangulation: 3D

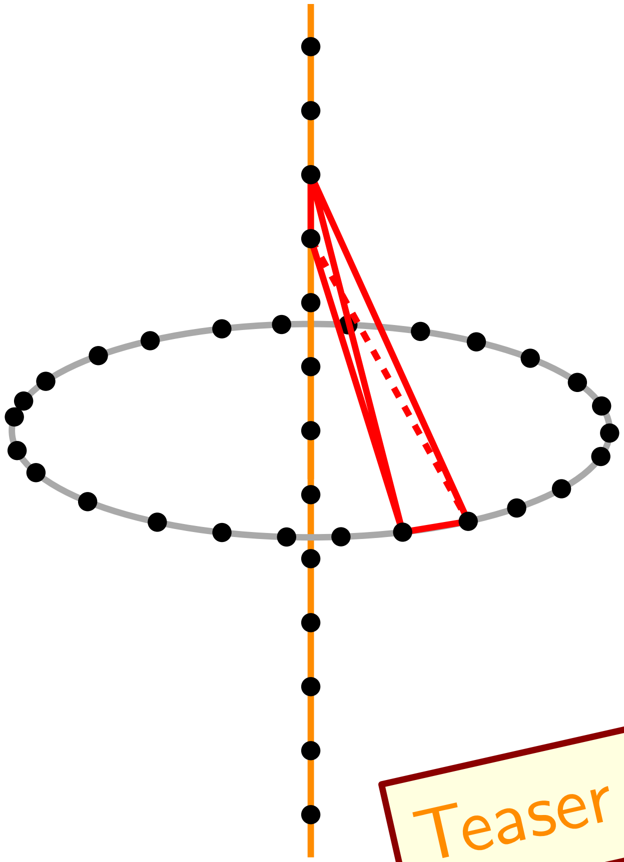
Quadratic examples

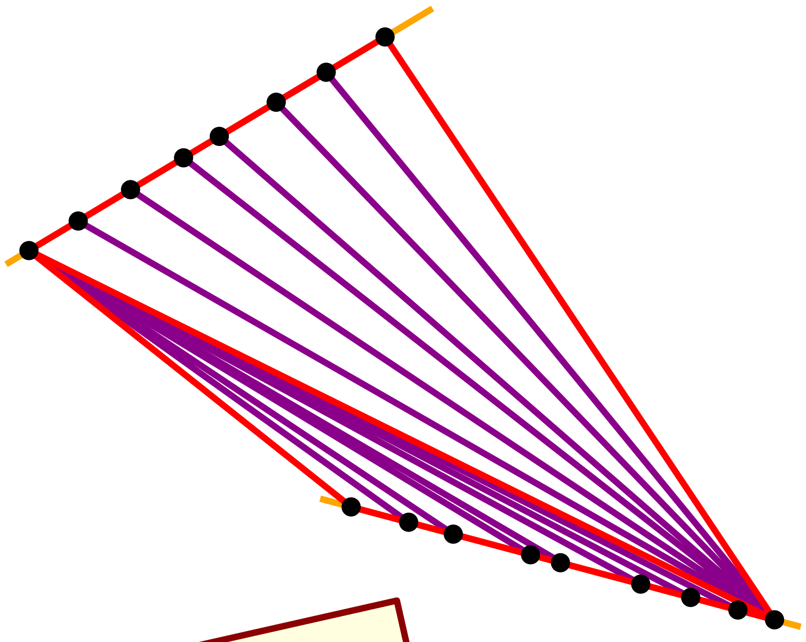




Delaunay Triangulation: 3D

Quadratic examples





Teaser probability lecture

Better results for random points

Delaunay Triangulation: 3D

Algorithms

4D convex hull duality

~~Flip~~

Incremental

Delaunay Triangulation: 3D

Algorithms

4D convex hull duality $O(f \log n + n^{\frac{4}{3}})$ or $\Theta(n^2)$

~~Flip~~

Incremental $\Theta(n^3)$

practical

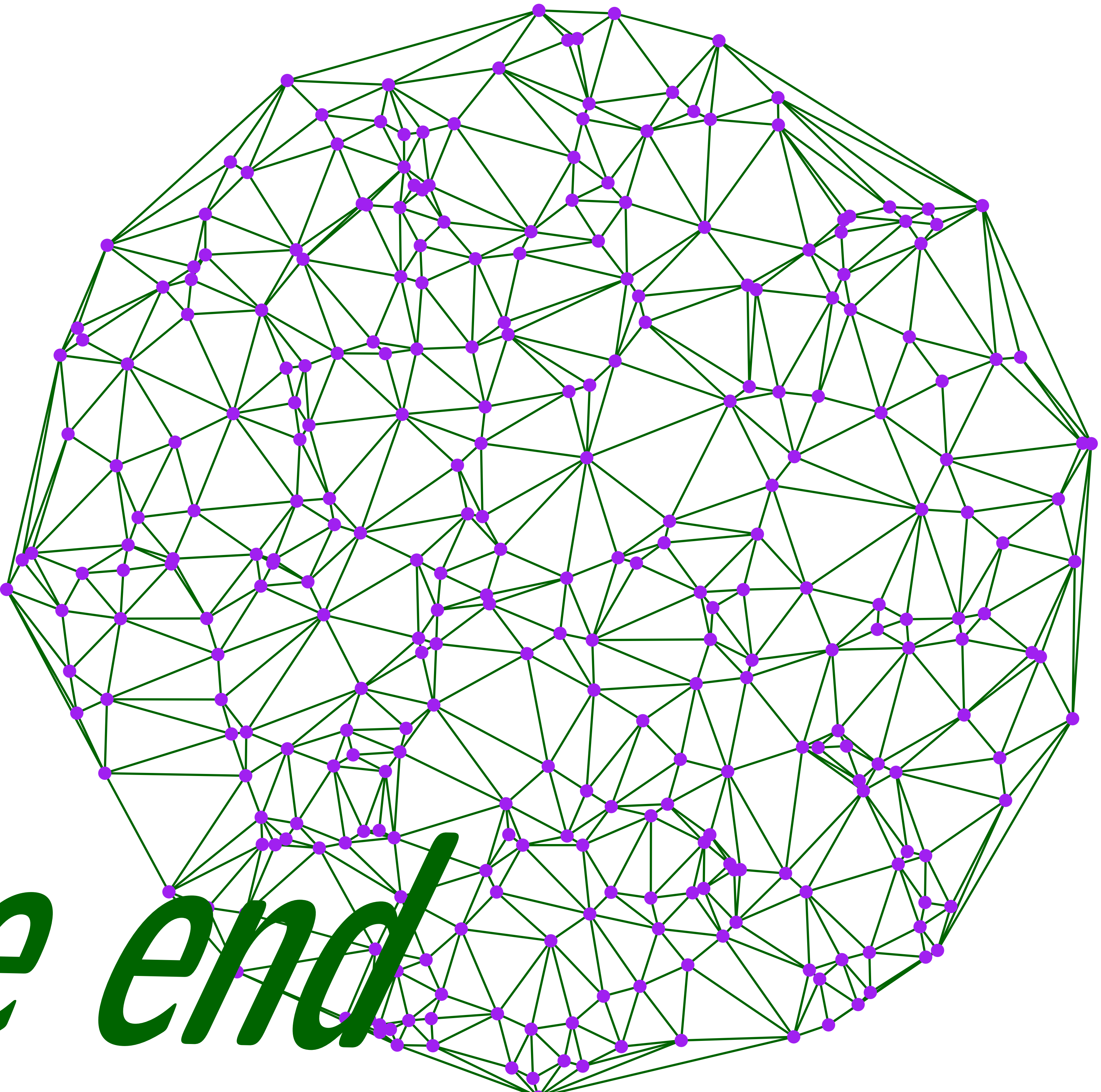
Teaser randomization lecture

Delaunay Triangulation: higher dimensions

$d+1$ convex hull duality $O\left(n^{\lfloor \frac{d+1}{2} \rfloor}\right)$

Incremental practical $O(n)$ for random points

coeff exponential in d



The end